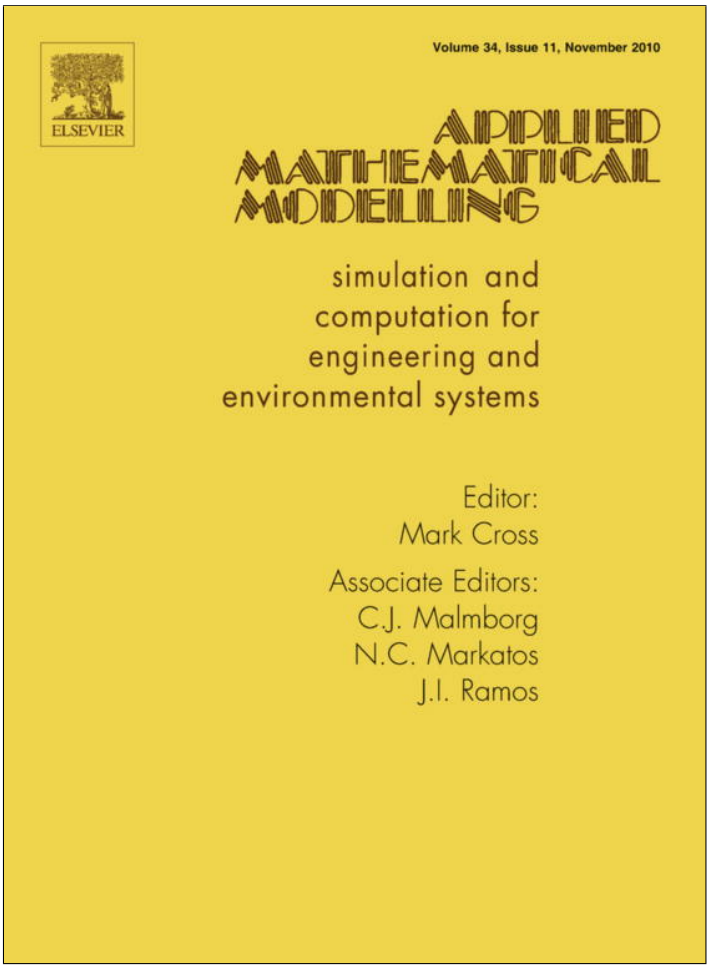

Applied Mathematical Modelling 34 (2010) 3547–3559

Contents lists available at ScienceDirect

Applied Mathematical Modelling

journal homepage: www.elsevier.com/locate/apm

Neuro-fuzzy based constraint programming

Hadi Sadoghi Yazdi [a,*], S.E. Hosseini [b], Mehri Sadoghi Yazdi [c]

[a] Computer Department, Ferdowsi University of Mashhad, Mashhad, Iran
[b] Engineering Department, Tarbiat Moallem University of Sabzevar, Sabzevar, Iran
[c] Electrical and Computer Engineering Department, Shahid Beheshti University, Tehran, Iran

ARTICLE INFO

Article history:
Received 21 April 2007
Received in revised form 8 March 2010
Accepted 10 March 2010
Available online 17 March 2010

Keywords:
Adaptive neuro-fuzzy inference system
Parametric programming
Constraint satisfaction
Learning
Fuzzy constraints
Monte-Carlo simulation

ABSTRACT

Constraint programming models appear in many sciences including mathematics, engineering and physics. These problems aim at optimizing a cost function joint with some constraints. Fuzzy constraint programming has been developed for treating uncertainty in the setting of optimization problems with vague constraints. In this paper, a new method is presented into creation fuzzy concept for set of constraints. Unlike to existing methods, instead of constraints with fuzzy inequalities or fuzzy coefficients or fuzzy numbers, vague nature of constraints set is modeled using learning scheme with adaptive neural-fuzzy inference system (ANFIS). In the proposed approach, constraints are not limited to differentiability, continuity, linearity; also the importance degree of each constraint can be easily applied. Unsatisfaction of each weighted constraint reduces membership of certainty for set of constraints. Monte-Carlo simulations are used for generating feature vector samples and outputs for construction of necessary data for ANFIS. The experimental results show the ability of the proposed approach for modeling constrains and solving parametric programming problems.

1. Introduction

Fuzzy logic systems can deal with incomplete knowledge of an environment with a vague nature. This makes fuzzy logic a powerful tool for reasoning in imprecise knowledge based systems. Due to continuous-valued logic nature, fuzzy logic is much closer to human thinking than the conventional two-valued logic. Problems with need to reasoning on imprecise knowledge are called fuzzy programming problems. Fuzzy programming problems have an outstanding relevance in many fields, such as those related to artificial intelligence, operations research. Therefore, many researches have been focused on solving these problems.

In [1], Herrera and Verdegay in 1995 presented methods to solve fuzzy integer linear programming (FILP) problems with different forms of fuzzy constraints, fuzzy numbers in the objective function and fuzzy numbers defining the set of constraints. They, in FILP problems with fuzzy constraints, defined constraints with fuzzy inequalities as follows:

$$\begin{aligned} &\text{Maximize} \quad z = \sum_{j=1}^{N} C_j x_j, \\ &\text{Subject to :} \quad \sum_{j=1}^{N} a_{ij} x_j \lesssim b_i, \quad i = 1, \ldots, M, \\ &\qquad x_j \geq 0 \quad j = 1, \ldots, N, \\ &\qquad x_j \text{ is integer}, \quad j = 1, \ldots, N, \end{aligned} \tag{1}$$

* Corresponding author.
E-mail address: h-sadoghi@um.ac.ir (H.S. Yazdi).

0307-904X/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.apm.2010.03.005

where M is the number of constraints and N is the number of variables to be optimized. They considered fuzzy constraints defined by membership functions:

$$\mu_i : R^n \to (0,1], \quad i = 1,\ldots,M, \tag{2}$$

and ith constraint corresponds to a membership function is as:

$$\mu_i(x) = \begin{cases} 1 & \text{if } \sum_{j=1}^{N} a_{ij}x_j \le b_i, frac[(b_i + d_i) - a_i x] d_i \text{if } b_i \le \sum_{j=1}^{N} a_{ij}x_j \le b_i + d_i, \\ 0 & \text{if } \sum_{j=1}^{N} a_{ij}x_j \ge b_i + d_i, \end{cases} \tag{3}$$

where $d_i > 0$, $i = 1,\ldots,M$. Then the problem is converted to a parametric programming with the parameter α-cut of the constraint $(\mu_X(x) \ge \alpha)$, where $X(\alpha) = \{x \in R^n | \mu_X(x) \ge \alpha\}$, $\forall x \in R^n$, $\mu_X(x) = \inf\{\mu_i(x), i \in M\}$. Therefore, each constraint in (1) can be written as:

$$\begin{aligned} &\text{Maximize} \quad z = \sum_{j=1}^{N} c_j x_j, \\ &\text{Subject to} \quad a_i x_j \le b_i + d_i(1-\alpha), \quad i = 1,\ldots,M, \\ &\qquad x_j \ge 0, \quad j = 1,\ldots,N, \\ &\qquad x_j \text{ is integer}, \quad j = 1,\ldots,N, \quad \alpha \in (0,1]. \end{aligned} \tag{4}$$

Also they studied FILP problems with imprecise coefficients in the objective function, that is, with coefficients defined by fuzzy numbers in the following form:

$$\begin{aligned} &\text{Maximize} \quad z = \sum_{j=1}^{N} \underset{\sim}{c}_j x_j, \\ &\text{Subject to} \quad \sum_{j=1}^{N} a_{ij}x_j \le b_i, \quad i = 1,\ldots,M, \\ &\qquad x_j \ge 0, \quad j = 1,\ldots,N, \\ &\qquad x_j \text{ is integer}, \quad j = 1,\ldots,N. \end{aligned} \tag{5}$$

Moreover, they studied a linear programming problem with fuzzy technological coefficients and fuzzy right hand-side numbers:

$$\begin{aligned} &\text{Maximize} \quad z = \sum_{j=1}^{N} c_j x_j, \\ &\text{Subject to} \quad \sum_{j=1}^{N} \underset{\sim}{a}_{ij} x_j < \underset{\sim}{b}_i, i \in M, \\ &\qquad x_j \ge 0, \quad j = 1,\ldots,N, \\ &\qquad x_j \text{ is integer}, \quad j = 1,\ldots,N. \end{aligned} \tag{6}$$

In [2], Rommelfanger used aggregation operator instead of addition in constraints which resulted in better fusion between components of each constraint, and constraints are close to real world. Jimenez and his team with companionship Verdegay focused on fuzzy nonlinear programming problems in which, as said, coefficients and/or constraints defining the problem are given as fuzzy ones [3]. Fuzzy solutions are obtained by means of a parametric approach in conjunction with evolutionary techniques. Moreover, an evolutionary algorithm was used as search algorithm to find optimum solution for nonlinear parametric programming problem.

In [4] Leon and Vercher presented a class of fuzzy linear programming problems in which coefficients in the constraints were modeled as LR-fuzzy numbers. Maity and Maiti [5] used fuzzy constraints for finding optimal production with the objective of minimum cost in the context of a multi-item dynamic production inventory control system. In their model constraints included fuzzy members. Mula and his cooperators tried to use fuzzy mathematical programming model for production planning under uncertainty in an industrial environment [6]. This model considered fuzzy constraints related to the total cost, the market demand and the available capacity of the productive resources and fuzzy coefficients for costs due to the backlog of demand and for the required capacity.

Here some notes must be mentioned about previous methods.

- If each converted constraint for one selected parameter is not satisfied, the answer is not accepted.
- Selected range of parameters in obtained parametric programming problem is questionable.

In the proposed method all constraint are substituted with fuzzy inference system which is tuned using neural network. In this system, if one of constraint is not satisfied only membership function of accuracy decrease toward zero. For the first time we use adaptive neural-fuzzy inference (ANFIS) system for expressing of constraints to form of humans like and think about constraints. Constraints are converted to rules with uncertainty and are mixed same as occur in a fuzzy system. In continue of introduction, we study recent work on ANFIS and will deduce new constraint learning using ANFIS is new work in the field of fuzzy constraint and ANFIS applications.

1.1. Short review on ANFIS application

Intelligent computing tools such as artificial neural network and fuzzy logic approaches are demonstrated to be competent when applied individually to a variety of problems. Recently, there has been a growing interest in combining both approaches, and as a result, neuro-fuzzy computing techniques have been evolved. ANFIS model combines the neural network adaptive capabilities and the fuzzy logic qualitative nature which we use it for constraint learning in the proposed method.

ANFIS was first presented by Jang [7]. It has attained its popularity due to a broad range of useful applications in such diverse areas in recent years as optimization of fishing predictions [8], vehicular navigation [9], identify the turbine speed dynamics [10], radio frequency power amplifier linearization [11], microwave application [12], image de-noising [13,14], prediction in cleaning with high pressure water [15], sensor calibration [16], fetal electrocardiogram extraction from ECG signal captured from mother [17], identification of normal and glaucomatous eyes from the quantitative assessment of summary data reports of the Stratus optical coherence tomography in Taiwan Chinese population [18]. These applications show that ANFIS is a good universal approximator, predictor, interpolator and estimator. They demonstrate that any nonlinear function of many inputs and outputs can be easily constructed with ANFIS.

This study aims at using ANFIS for constraints learning in a convex or non-convex optimization problem (see Appendix). Learning of constraints leads to substitution of crisp constraints with a fuzzy system tuned with neural networks. Many problems exist in constraints satisfaction problems which are resolved due to using ANFIS model. In conventional optimization methods, unsatisfying of any constraint causes searching procedure hit a change in search trace, whereas in the proposed model, unsatisfaction of each constraint reduces membership of certainty for set of constraints which is a type of penalty for cost function. In our method, all constraints are substituted with fuzzy inference system (FIS) which is tuned using neural network. In this system, if one of constraints is not satisfied only membership function of accuracy decreases toward zero. For the first time, we use adaptive neural-fuzzy inference system (ANFIS) for expressing constraints to real form that human think. Constraints are converted to rules with uncertainty and are mixed just the same as occur in a fuzzy system.

The paper is organized as follows. The architecture of adaptive neuro-fuzzy inference system is explained in Section 2. Section 3 is devoted to solving constraint parametric programming using ANFIS algorithm. Experimental results are discussed in Section 4 and in final section conclusions are presented.

2. Adaptive neuro-fuzzy inference system (ANFIS) architecture

Neural networks (NNs) are demonstrated to have powerful capability of expressing relationship between input–output variables. In fact it is always possible to develop a structure that approximates a function with a given precision. However, there is still distrust about NNs identification capability in some applications [19]. Fuzzy set theory plays an important role in dealing with uncertainty in plant modeling applications. Neuro-fuzzy systems are fuzzy systems, which use NNs to determine their properties (fuzzy sets and fuzzy rules) by processing data samples [20]. Neuro-fuzzy integrates to synthesize the merits of both NN and fuzzy systems in a complementary way to overcome their disadvantages. The fusion of an NN and fuzzy logic in neuro-fuzzy models possess both low-level learning and computational power of NNs and advantages of high-level human like thinking of fuzzy systems. For identification, hybrid neuro-fuzzy system called ANFIS combines a NN and a fuzzy system together. ANFIS has been proved to have significant results in modeling nonlinear functions. In ANFIS, the membership functions (MF) are extracted from a data set that describes the system behavior. The ANFIS learns features in the data set and adjusts the system parameters according to given error criterion. In a fused architecture, NN learning algorithms are used to determine the parameters of fuzzy inference system. Below, we have summarized the advantages of the ANFIS technique.

- Real-time processing of instantaneous system input and output data's. This property helps using of this technique for many operational researches problems.
- Offline adaptation instead of online system-error minimization, thus easier to manage and no iterative algorithms are involved.
- System performance is not limited by the order of the function since it is not represented in polynomial format.
- Fast learning time.
- System performance tuning is flexible as the number of membership functions and training epochs can be altered easily.
- The simple if–then rules declaration and the ANFIS structure are easy to understand and implement.

A typical architecture of ANFIS is shown in Fig. 1, in which a circle indicates a fixed node, and a square indicates an adaptive node. For simplicity, we consider two inputs x, y and one output z in the FIS. The ANFIS used in this paper implements a first-order Sugeno fuzzy model. Among many FIS models, the Sugeno fuzzy model is the most widely used for its high interpretability and computational efficiency, and built-in optimal and adaptive techniques. For a first-order Sugeno fuzzy model, a common rule set with two fuzzy if–then rules can be expressed as:

$$\text{Rule 1 : If x is A1 and y is B1, then}$$
$$z_1 = p_1 x + q_1 y + r_1, \tag{7}$$
$$\text{Rule 2 : If x is A2 and y is B2, then}$$
$$z_2 = p_2 x + q_2 y + r_2, \tag{8}$$

where A_i, B_i $(i = 1, 2)$ Ai and Bi are fuzzy sets in the antecedent, and p_i, q_i, r_i $(i = 1, 2)$ are the design parameters that are determined during the training process. As in Fig. 1, the ANFIS consists of five layers:

Layer 1, every node i in this layer is an adaptive node with a node function:

$$\begin{aligned} O_i^1 &= \mu_{A_i}(x), \quad i = 1, 2, \\ O_i^1 &= \mu_{B_i}(y), \quad i = 3, 4, \end{aligned} \tag{9}$$

where x, y are the input of node i, and $\mu_{A_i}(x)$ and $\mu_{B_i}(y)$ can adopt any fuzzy membership function (MF). In this paper, Gaussian MFs are used:

$$gaussian(x, c, \sigma) = e^{-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2}, \tag{10}$$

where c is center of Gaussian membership function and σ is standard deviation of this cluster.

Layer 2, every node in the second layer represents the ring strength of a rule by multiplying the incoming signals and forwarding the product as:

$$O_i^2 = \omega_i = \mu_{A_i}(x)\mu_{B_i}(y), \quad i = 1, 2. \tag{11}$$

Layer 3, the ith node in this layer calculates the ratio of the ith rule's ring strength to the sum of all rules' ring strengths:

$$O_i^3 = \varpi_i = \frac{\omega_i}{\omega_1 + \omega_2}, \quad i = 1, 2, \tag{12}$$

where ϖ_i is referred to as the normalized ring strengths.

Layer 4, the node function in this layer is represented by:

$$O_i^4 = \varpi_i z_i = \varpi_i (p_i x + q_i y + r_i), \quad i = 1, 2, \tag{13}$$

where ϖ_i is the output of layer 3, and $\{p_i, q_i, r_i\}$ are the parameter set. Parameters in this layer are referred to as the consequent parameters.

Layer 5, the single node in this layer computes the overall output as the summation of all incoming signals:

$$O_i^5 = \sum_{i-1}^{2} \varpi_i z_i = \frac{\omega_1 z_1 + \omega_2 z_2}{\omega_1 + \omega_2}. \tag{14}$$

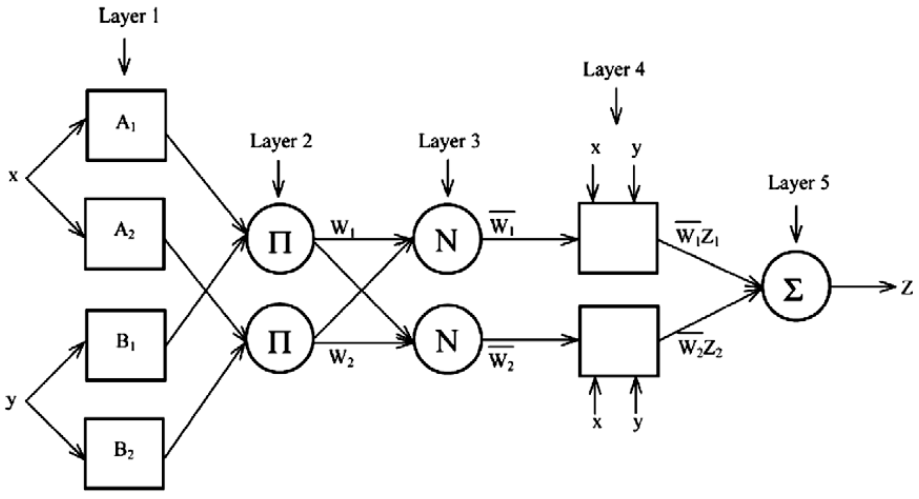

Fig. 1. ANFIS architecture. Π, N, $\sum$ are defined in (11), (12), (14), respectively.

It is seen from the ANFIS architecture that when the values of the premise parameters are fixed, the overall output can be expressed as a linear combination of the consequent parameters:

$$z = (\varpi_1 x)p_1 + (\varpi_1 y)q_1 + (\varpi_1)r_1 + (\varpi_2 X)P_2 + (\varpi_2 y)q_2 + (\varpi_2)r_2. \tag{15}$$

The hybrid learning algorithm [7,21] combining the least square method and the back propagation (BP) algorithm can be used to solve this problem. This algorithm converges much faster since it reduces the dimension of the search space of the BP algorithm. During the learning process, the premise parameters in layer 1 and the consequent parameters in layer 4 are tuned until the desired response of the FIS is achieved. The hybrid learning algorithm has a two-step process. First, while holding the premise parameters fixed, the functional signals are propagated forward to layer 4, where the consequent parameters are identified by the least square method. Second, the consequent parameters are held fixed while the error signals, the derivative of the error measure with respect to each node output, are propagated from the output end to the input end, and the premise parameters are updated by the standard BP algorithm.

3. The proposed method

Let us consider the following parametric programming problem:

$$\begin{aligned} &\text{Minimize} \quad f(x,\theta), \\ &\text{Subject to} \quad h_i(x,\theta) \le 0, \quad i = 1,\ldots,m, \\ &\qquad\qquad \theta \in [\alpha,\beta], \end{aligned} \tag{16}$$

α, β are constants and $x \in R^n$.

Here $f(x,\theta)$ and $h_i(x,\theta)$ $(i = 1,\ldots,m)$ are functions with respect to the first argument and $\theta \in [\alpha,\beta]$. Constraints $h_i(x,\theta)$ may be non-differentiable, discontinuous and nonlinear.

In conventional method for solving parametric programming problems or fuzzy parametric programming problems which is converted to a parametric programming problem, if each constraint is not satisfied x must be deleted from answer set. But in the proposed method, value of unsatisfaction of each constraints decrease membership of certainty for set of constraints or increase penalty value to cost function $f(x,\theta)$ to linear (Fig. 2) or nonlinear form. Value of penalty and its relation can be defined with user.

In the proposed approach, firstly set of x, θ are generated and $h_i(x,\theta)$ $(i = 1,\ldots,m)$ are observed which this work is performed using Monte-Carlo simulation (see Appendix). Obtained values for each constraint $h_i(x,\theta)$ $(i = 1,\ldots,m)$ are points in m-dimensional space. Penalty values may be defined to linear or nonlinear form versus $h_i(x,\theta)$ $(i = 1,\ldots,m)$ depend on user desideration. Second step in the proposed method is fitting procedure over penalty term versus constraints which is performed using ANFIS. In the following section, complete explanation is presented.

3.1. Constraint learning by ANFIS

In this work, the ANFIS is used to learn constrains for solving a parametric programming problem. We define a feature vector as input to ANFIS which each feature includes generated value of one constraint; therefore the length of the feature vector is m. The feature vector for kth sample is as follows:

$$\vec{F}_k = \left\{\hat{h}_1(x,\theta), \hat{h}_2(x,\theta), \ldots, \hat{h}_m(x,\theta)\right\}_k \quad k = 1,\ldots,N, \tag{17}$$

where

$$\hat{h}_i(x,\theta) = \begin{cases} h_i(x,\theta) & h_i(x,\theta) > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{18}$$

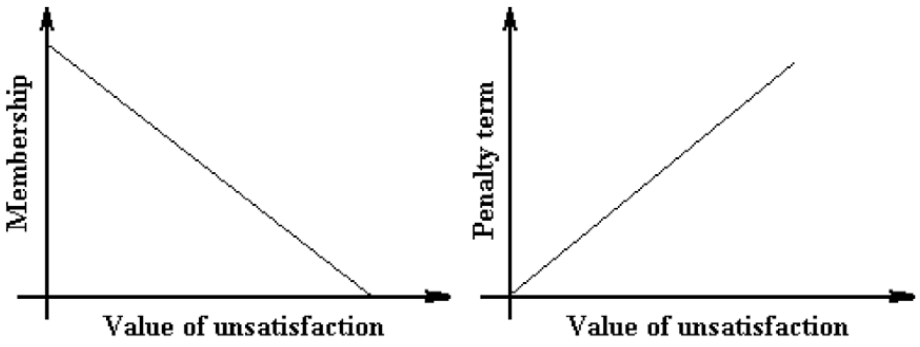

Fig. 2. Membership of certainty for set of constraints and penalty value versus value of satisfaction to linear form.

And $\vec{F}_k$'s are feature vectors for N training/generated samples. $\hat{h}_i(x,\theta)$ is penalty value and in this paper is selected to form of (18). It can be defined to nonlinear form. Monte-Carlo simulation is used for generating N samples of feature vectors. Also for kth sample, desired output with linear form is defined as follows. Of course in the general form desired output is given by user

$$\vec{d}_k = \frac{\sum_{i=1}^{m} \alpha_i |\hat{h}_i(x,\theta)|}{\max_i \{\alpha_i \hat{h}_i(x,\theta)\}}, \quad k = 1,\ldots,N, \quad i = 1,\ldots,M, \tag{19}$$

where α_i is the weight of each constraint. Denominator of (19) is a normalization factor and its numerator is linear combiner of constraints so the desired output is a linear function of constraints, albeit constraints can be nonlinear functions of x and θ. Of course, α_i determine the slop of interpolated hyperplane. If a constraint $h_i(x,\theta)$ is not satisfied, $\vec{d}_k$ increases proportional to $|h_i(x,\theta)|$, as a penalty value. This penalty affects the output with selected slop of α_i. If each constraint is satisfied, then the related penalty is zero. A fuzzy inference system is generated with $\vec{F}_k$ and $\vec{d}_k$, and tuned using neural network which is mentioned in Section 2. Obtained system has properties of ANFIS system is to following form:

$$g(x,\theta) = N(\vec{F}_k, \vec{d}_k). \tag{20}$$

$g(x,\theta)$ is continuously differentiable and linear whereas constraints do not have these properties. $N(\vec{F}_k, \vec{d}_k)$ is the constructed fuzzy inference system. Now problem (16) is transformed to a conventional optimization model (21).

$$\text{minimize}_x \; p(x,\theta) = f(x,\theta) + \eta g(x,\theta) \quad \theta \in [\alpha,\beta], \tag{21}$$

where η is a positive number. We can solve this problem with different methods, like optimization methods or neural network based method [22].

4. Experimental results

Three examples are considered as follows.

Example 1. Consider the following convex parametric problem:

$$\begin{aligned} \text{Minimize} \quad & z(\theta) = (\theta^2 - \theta - 3)x_1 + (\theta^2 + 2\theta - 6)x_2, \\ \text{Subject to} \quad & x_1 \le 4 + \theta^2, \\ & 3x_1 + 2x_2 \le 18 - 2\theta^2, \\ & x_1, x_2 \ge 0, \\ & \theta \in [0,3]. \end{aligned} \tag{22}$$

Firstly, constraints are converted to ANFIS model. For this purpose we have:

$$\hat{h}_1(x,\theta) = \begin{cases} 4 + \theta^2 - x_1 & \text{if } 4 + \theta^2 - x_1 > 0 \\ 0 & \text{elsewhere} \end{cases}, \tag{23}$$

$$\hat{h}_2(x,\theta) = \begin{cases} 18 - 2\theta^2 - 3x_1 - 2x_2 & 18 - 2\theta^2 - 3x_1 - 2x_2 > 0 \\ 0 & \text{elsewhere} \end{cases}, \tag{24}$$

$$\hat{h}_3(x,\theta) = \begin{cases} x_1 & x_1 > 0 \\ 0 & \text{elsewhere} \end{cases}, \tag{25}$$

$$\hat{h}_4(x,\theta) = \begin{cases} x_2 & x_2 > 0 \\ 0 & \text{elsewhere} \end{cases}. \tag{26}$$

Then $\vec{F}_k = \{\hat{h}_1(x,\theta), \hat{h}_2(x,\theta), \hat{h}_3(x,\theta), \hat{h}_4(x,\theta)\}_k$, $k = 1,\ldots,1000$ is constructed using Monte-Carlo simulation, x_1, x_2 and θ are computed according to:

$$x_1 = \beta_1(v_1 - 0.5), \quad x_2 = \beta_2(v_2 - 0.5), \quad \theta = \beta_3(v_3 - 0.5), \tag{27}$$

where β_1, β_2, β_3 determine the range of variables, and v_1, v_2, v_3 are random numbers. Random variables are between (0, 1), so x_1, x_2, θ are between $(-0.5\,\beta_1, 0.5\,\beta_1)$, $(-0.5\,\beta_2, 0.5\,\beta_2)$, $(-0.5\,\beta_3, 0.5\,\beta_3)$, respectively.

In this example β_1, β_2, β_3 are selected equal to 20. Then $\vec{d}_k$ is obtained according to (19), with α_i equal to one, and ANFIS model of $N(\vec{F}_k, \vec{d}_k)$ is generated. Obtained input membership functions are shown in Fig. 3 for 2nd constraint $\hat{h}_2(x,\theta)$. This figure shows this constraint is not satisfied with values between −60 to 0, of course Monte-Carlo simulation lets all states to be checked. Output membership function is obtained as a linear function of input constraints, as described in Section 2 and as Eq. (28) shows:

$$z_k = -0.0041 \sum_{i=1}^{4} \hat{h}_i(x,\theta), \quad k = 1,\ldots,4. \tag{28}$$

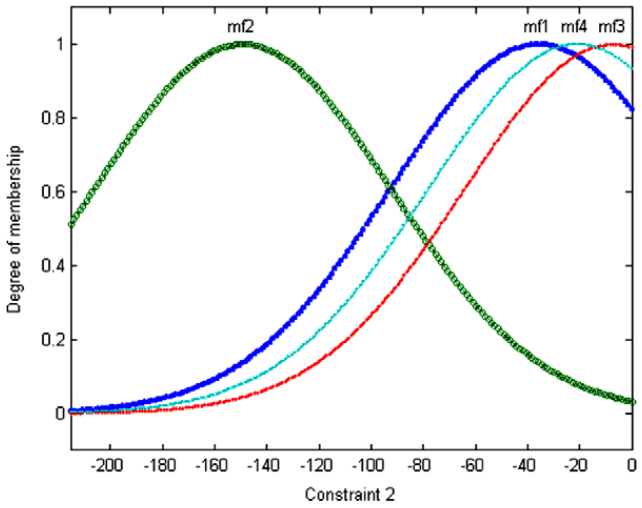

Fig. 3. Obtained input membership functions for 2nd constraint $\hat{h}_2(x,\theta)$, mf1 (or $Mf_1(\hat{h}_2(x,\theta))$), mf2 (or $Mf_2(\hat{h}_2(x,\theta))$), mf3 (or $Mf_3(\hat{h}_2(x,\theta))$) are label of membership functions.

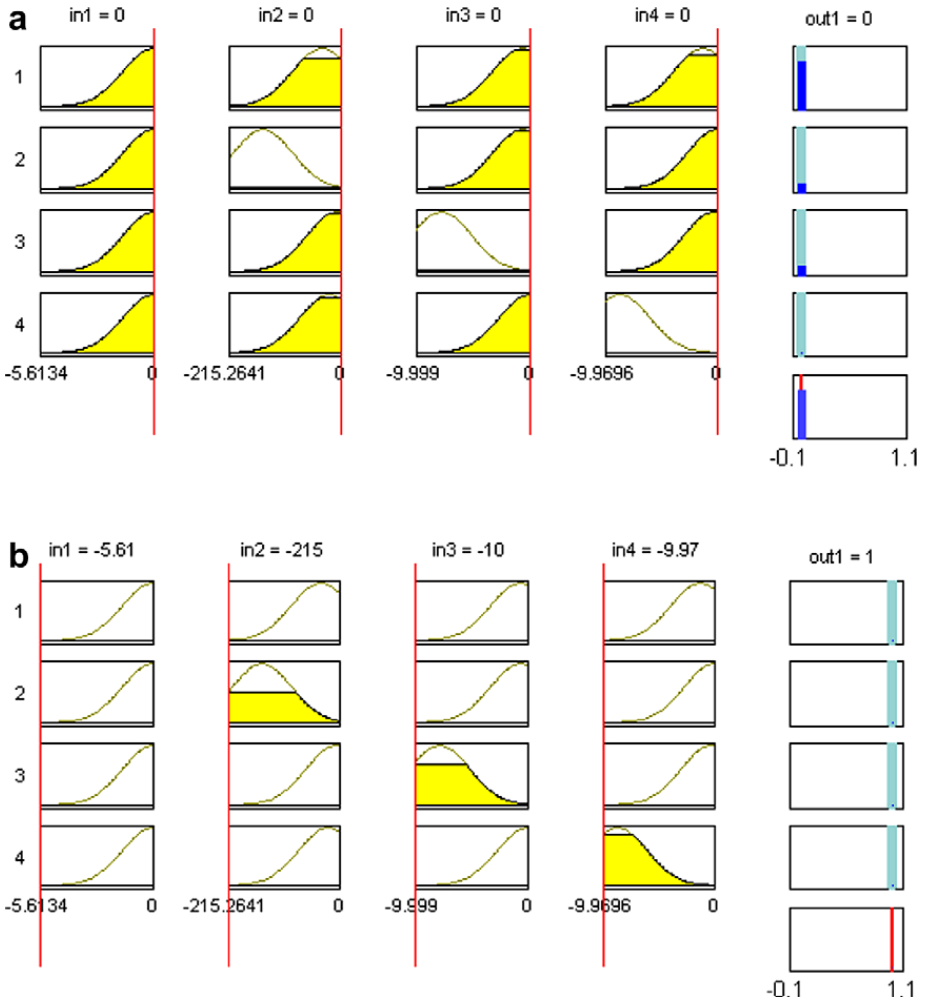

Fig. 4. Rule operation when constrains are selected extreme values. in1, in2, in3, in4 are four constraints $\hat{h}_1(x,\theta)$, $\hat{h}_2(x,\theta)$, $\hat{h}_3(x,\theta)$, $\hat{h}_4(x,\theta)$. (a) Constrains are satisfied and penalty term (Out1) is zeros. (b) All constraints are not satisfied (worst case) and penalty term is maximum.

Generated rules are,

1. If $\hat{h}_1(x,\theta)$ is $Mf_1(\hat{h}_1(x,\theta))$ and $\hat{h}_2(x,\theta)$ is $Mf_1(\hat{h}_2(x,\theta))$ and $\hat{h}_3(x,\theta)$ is $Mf_1(\hat{h}_3(x,\theta))$ and $\hat{h}_4(x,\theta)$ is $Mf_1(\hat{h}_4(x,\theta))$. Then Output is z_1.

2. If $\hat{h}_1(x,\theta)$ is $Mf_2(\hat{h}_1(x,\theta))$ and $\hat{h}_2(x,\theta)$ is $Mf_2(\hat{h}_2(x,\theta))$ and $\hat{h}_3(x,\theta)$ is $Mf_2(\hat{h}_3(x,\theta))$ and $\hat{h}_4(x,\theta)$ is $Mf_2(\hat{h}_4(x,\theta))$. Then Output is z_2.
3. If $\hat{h}_1(x,\theta)$ is $Mf_3(\hat{h}_1(x,\theta))$ and $h_2(x,\theta)$ is $Mf_3(\hat{h}_2(x,\theta))$ and $\hat{h}_3(x,\theta)$ is $Mf_3(\hat{h}_3(x,\theta))$ and $\hat{h}_4(x,\theta)$ is $Mf_3(\hat{h}_4(x,\theta))$. Then Output is z_3.
4. If $\hat{h}_1(x,\theta)$ is $Mf_4(\hat{h}_1(x,\theta))$ and $\hat{h}_2(x,\theta)$ is $Mf_4(\hat{h}_2(x,\theta))$ and $\hat{h}_3(x,\theta)$ is $Mf_4(\hat{h}_3(x,\theta))$ and $\hat{h}_4(x,\theta)$ is $Mf_4(\hat{h}_4(x,\theta))$. Then Output is z_4.

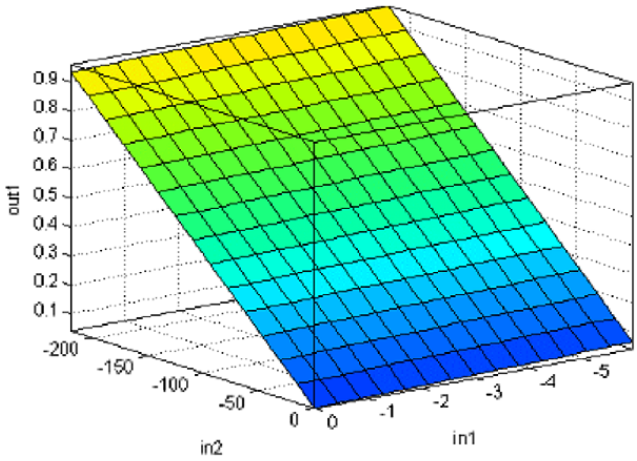

Fig. 5. Out1 is output, In1 is $\hat{h}_1(x,\theta)$, In2 is $\hat{h}_2(x,\theta)$; output per two constraints for Example 1.

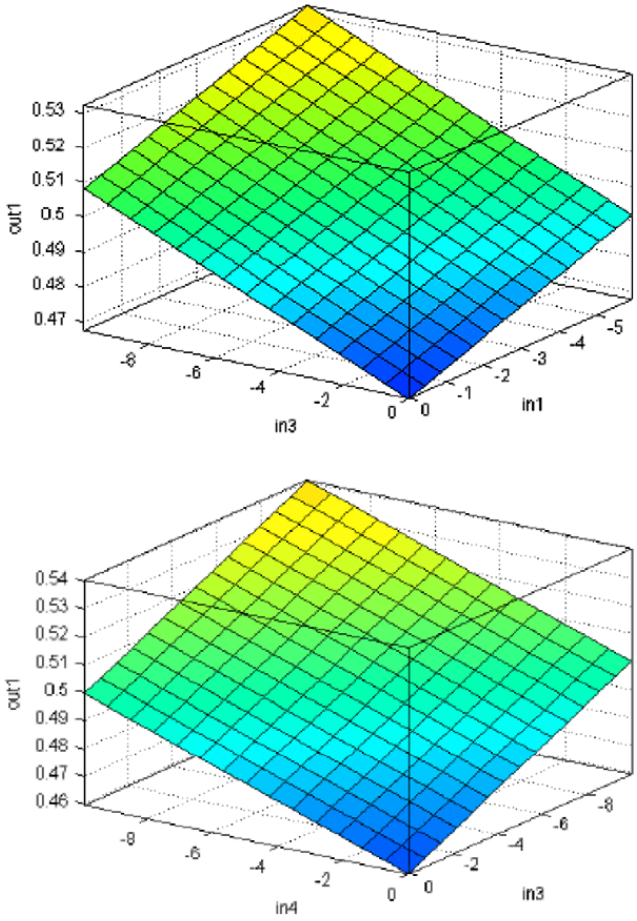

Fig. 6. Out1 is output, In1 is $\hat{h}_1(x,\theta)$, In2 is $\hat{h}_2(x,\theta)$, In3 is $\hat{h}_3(x,\theta)$, In4 is $\hat{h}_4(x,\theta)$; output per two constraints.

Constraints fire rules as shown in Fig. 4. This figure shows two states of rule operation for boundary conditions.

The importance degree of each constraint can be clearly seen from Fig. 5. If constraint $\hat{h}_2(x,\theta)$ is not satisfied, big penalty value is generated and search algorithm moves towards satisfaction of this constraint. But other constraints have minor

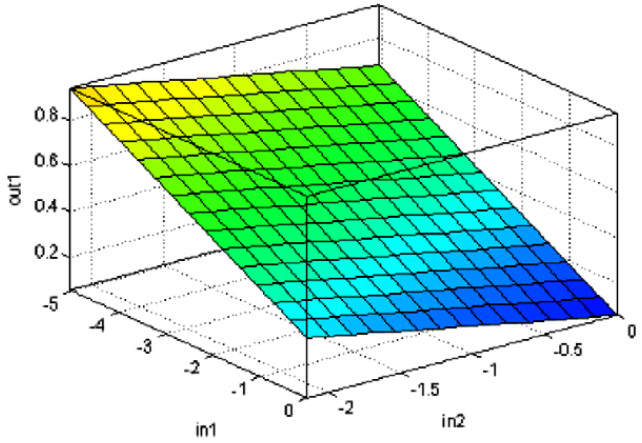

Fig. 7. Studying of effect of weights in constraints. For selected state $\alpha_1 = 0.9,\ \alpha_2 = 0.01,\ \alpha_3 = 0.05,\ \alpha_4 = 0.04$. Out1 is output, In1 is $\hat{h}_1(x,\theta)$, In2 is $\hat{h}_2(x,\theta)$; output per two constraints.

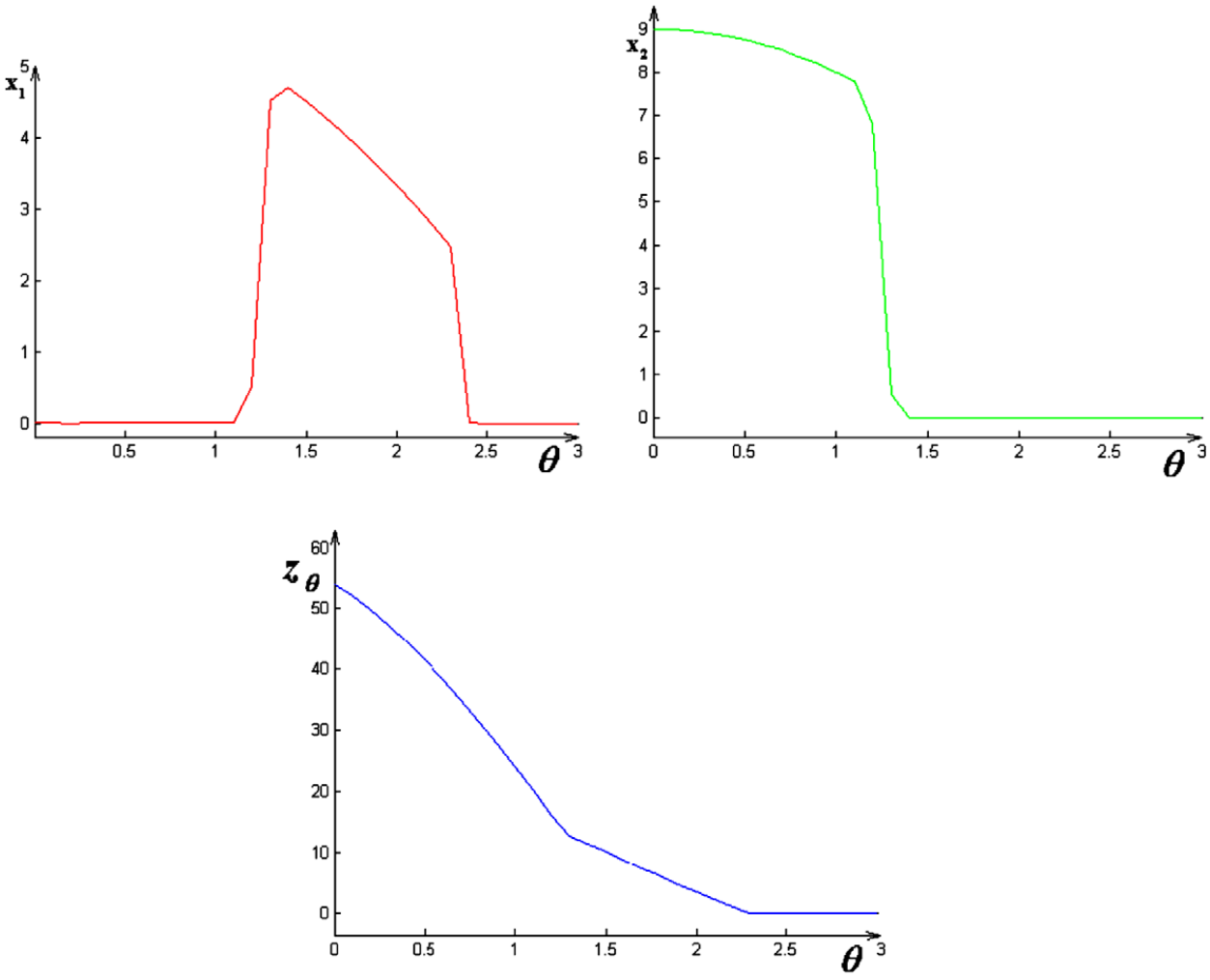

Fig. 8. Optimal solution for Example 1.

emphasis ($\{\hat{h}_1(x,\theta),\hat{h}_3(x,\theta),\hat{h}_4(x,\theta)\}$). This can be seen in Fig. 6, which output or same penalty value is shown for these constraints. An interesting note in these figures is the linearity of output in inputs, in spite of nonlinearity of constraints in (x, θ).

If one needs to increase the importance of one constraint, this is easily implemented in the proposed approach. For example in above example, we can increase the effect of the first constraint with a new weights selection, for example, $\alpha_1 = 0.9$, $\alpha_2 = 0.01$, $\alpha_3 = 0.05$, $\alpha_4 = 0.04$. Part of the results is shown in Fig. 7. This figure shows that unsatisfaction of first constraint causes considerable penalty value relative to 2nd constraint with normal effect.

Above problem is solved for quantized θ with resolution 0.1 i.e., $\theta \in \{0, 0.1, 0.2, \ldots, 2.9, 3.0\}$ include 31 solution points. The optimal solution is shown in Fig. 7 (see Fig. 8).Example 2 Consider the following convex parametric programming problem:

$$\begin{aligned} &\text{Minimize} && z(\theta) = -x_1^2 - 3x_2^2, \\ &\text{Subject to} && x_1 + x_2 \le 6 - \theta, \\ & && -x_1 + 2x_2 \le 6 + \theta, \\ & && x_1, x_2 \ge 0, \\ & && \theta \in [0,4]. \end{aligned} \tag{29}$$

The obtained solution using the proposed method is shown in Fig. 9. Importance or efficacy of constraints can be seen in Fig. 10. Constraints x_1, $x_2 \geqslant 0$ are same effect that have not illustrated in this figure. The efficacy of the first and the second constraints are shown in this figure. These figure shows that these constraints have approximately the same effect.

In our method for modeling constraints, the effect of each constraint can be studied. This method can be used for expressing constraints in the fuzzy form but outward show can be crisp or fuzzy.

For more emphasis on linearity of defined relation (19) between membership of penalty value and constraints, we study following example which have nonlinear constrains.

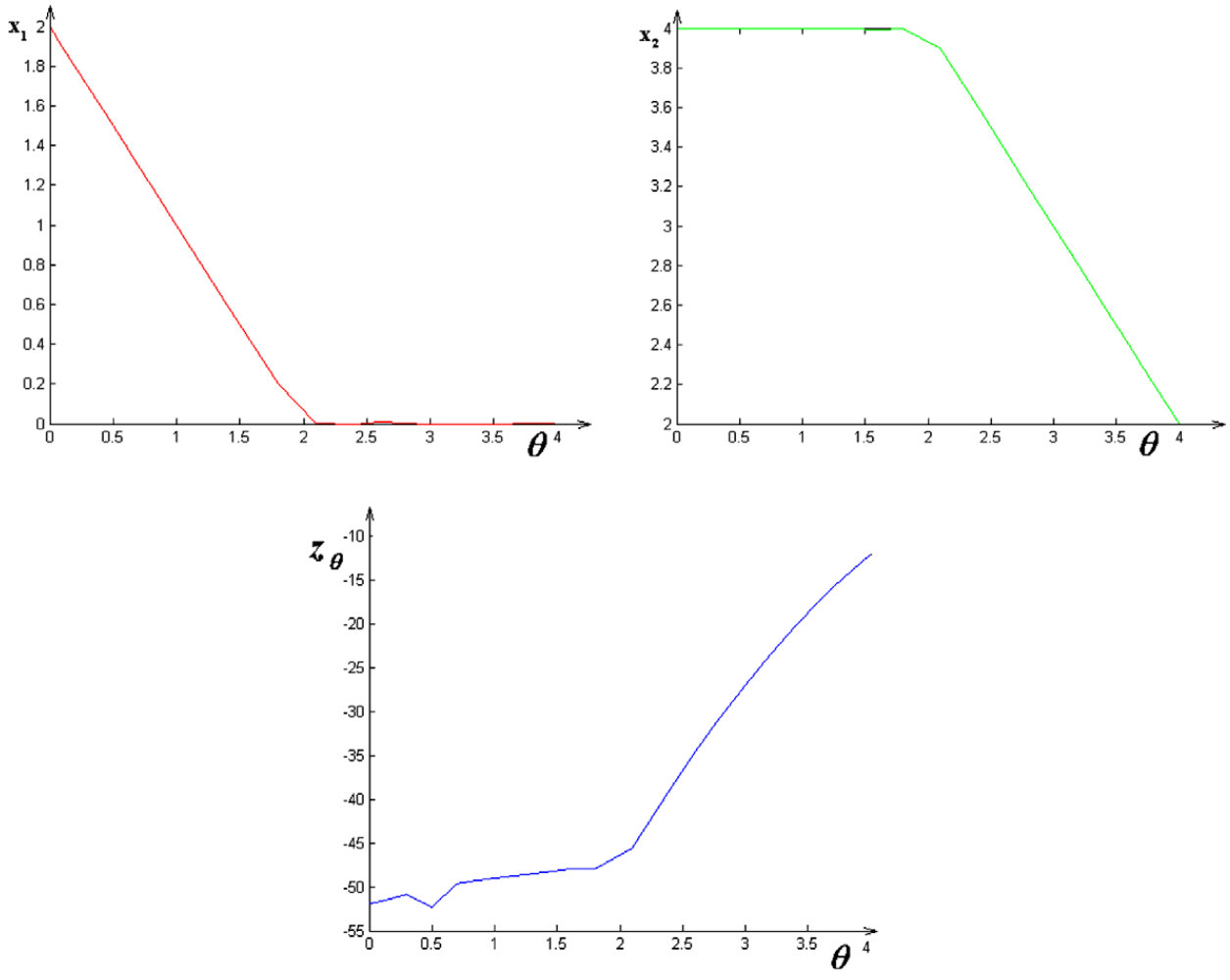

Fig. 9. Optimal solution for Example 2.

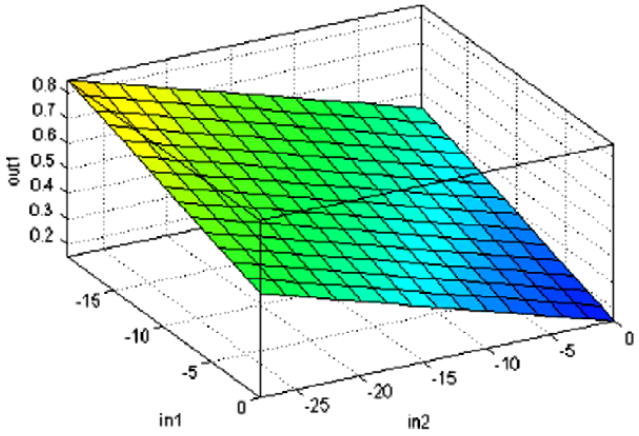

Fig. 10. Out1 is output, In1 is $\hat{h}_1(x,\theta)$, In2 is $\hat{h}_2(x,\theta)$; output per two constraints for Example 2.

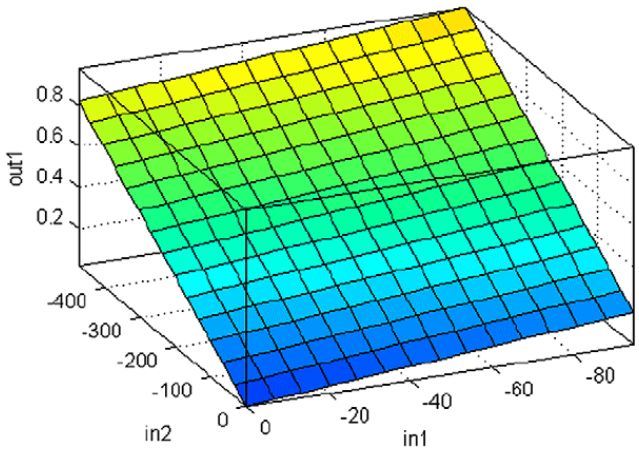

Fig. 11. Output of ANFIS after constraints learning in Example 3.

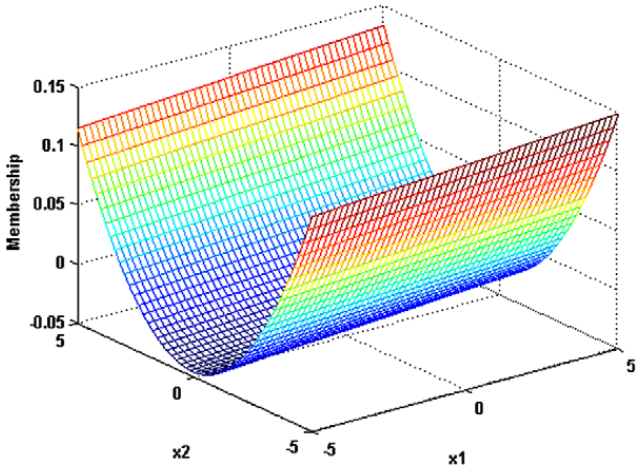

Fig. 12. Membership of penalty value per x_1, x_2.

Example 3. Consider the following constraints in one parametric programming problem:*Constraints:*

$$x_1^2 \le 4 + \theta^2,$$
$$3x_1^2 + 2x_2 \le 18 - 2\theta^2,$$
$$x_1, x_2 \ge 0,$$
$$\theta \in [0, 3].$$

The output of ANFIS after constraints learning using Monte-Carlo simulation (membership of penalty value) is shown in Fig. 11. We can easily see that a linear relation there exists between the membership penalty value of constrains which is pre-typified but nonlinearity of membership of penalty value and x_1, x_2 is perceptible and is shown in Fig. 12. Therefore, one of suitability of the proposed approach in constraint learning is that the effect of nonlinear constraints appears in the form of linear penalty and better convergence is achieved.

5. Conclusion

In this paper we studied a new model for dealing with the lack of precision in a vague nature in the formulation of parametric programming problems. The foundation of this model was based on constraint learning using adaptive neuro-fuzzy inference system. Simulation results show that each parametric programming problems can be trained using ANFIS. Understanding of constraints after generation of ANFIS model was one of sensible features.

Appendix

A.1. Convex optimization

Convex optimization is a sub-field of mathematical optimization. Given a real vector space X together with a convex, real valued function $f : X \to \mathbb{R}$ defined on a convex subset X of X, the problem is to find the point x^* in X for which the number $f(x)$ is smallest, i.e., the point x^* such that $f(x^*) \leqslant f(x)$ for all $x^* \in X$. The convexity of X and f make the powerful tools of convex analysis applicable: the Hahn–Banach theorem and the theory of sub-gradients lead to a particularly satisfying and complete theory of necessary and sufficient conditions for optimality, a duality theory comparable in perfection to that for linear programming, and effective computational methods.

A.2. Monte Carlo simulation

Monte Carlo methods are a class of computational algorithms that rely on repeated random sampling to compute their results. Monte Carlo methods are often used when simulating physical and mathematical systems. Because of their reliance on repeated computation and random or pseudo-random numbers, Monte Carlo methods are most suited to calculation by a computer. Monte Carlo methods tend to be used when it is infeasible or impossible to compute an exact result with a deterministic algorithm. Monte Carlo simulation methods are especially useful in studying systems with a large number of coupled degrees of freedom. More broadly, Monte Carlo methods are useful for modeling phenomena with significant uncertainty in inputs. These methods are also widely used in mathematics. The term Monte Carlo method was coined in the 1940s by physicists working on nuclear weapon projects in the Los Alamos National Laboratory.

References

[1] F. Herrera, J.L. Verdegay, Three models of fuzzy integer linear programming, Eur. J. Oper. Res. 83 (1995) 581–593.
[2] H. Rommelfanger, Fuzzy linear programming and applications, Eur. J. Oper. Res. 92 (1996) 512–527.
[3] F. Jimenez, J.M. Cadenas, J.L. Verdegay, G. S_anchez, Solving fuzzy optimization problems by evolutionary algorithms, Inform. Sci. 152 (2003) 303–311.
[4] T. Leon, E. Vercher, Solving a class of fuzzy linear programs by using semi-infinite programming techniques, Fuzzy Set. Syst. 146 (2004) 235–252.
[5] K. Maity, M. Maiti, Possibility and necessity constraints and their defuzzification – a multi-item production-inventory scenario via optimal control theory, Eur. J. Oper. Res. 177 (2007) 882–896.
[6] J. Mula, R. Polera, J.P. Garcia-Sabater, Material requirement planning with fuzzy constraints and fuzzy coefficients, Fuzzy Set. Syst. (2006), doi:10.1016/j.fss.2006.11.003.
[7] J.-S.R. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Syst. Man Cyb. 23 (5) (1993) 665–685.
[8] A.I. Nuno, B. Arcay, J.M. Cotos, J. Varela, Optimization of fishing predictions by means of artificial neural networks, ANFIS, functional networks and remote sensing images, Expert Syst. Appl. 29 (2005) 356–363.
[9] A. Noureldin, A. El-Shafie, M.R. Tahab, Optimizing neuro-fuzzy modules for data fusion of vehicular navigation systems using temporal cross-validation, Eng. Appl. Artif. Intel. 20 (2007) 49–61.
[10] N. Kishor, S.P. Singh, A.S. Raghuvanshic, Adaptive intelligent hydro turbine speed identification with water and random load disturbances, Eng. Appl. Artif. Intel. (2007), doi:10.1016/j.engappai.2006.11.014.
[11] K.C. Lee, P. Gardner, Adaptive neuro-fuzzy inference system (ANFIS) digital predistorter for RF power amplifier linearization, IEEE Trans. Veh. Technol. 55 (1) (2006) 43–51.
[12] E.D. Ubeyli, I. Guler, Adaptive neuro-fuzzy inference system to compute quasi-TEM characteristic parameters of micro shield lines with practical cavity sidewall profiles, Neurocomputing 70 (2006) 296–304.
[13] H. Qin, S.X. Yang, Adaptive neuro-fuzzy inference systems based approach to nonlinear noise cancellation for images, Fuzzy Set. Syst. (2007), doi:10.1016/j.fss.2006.10.028.
[14] P. Çivicioglu, Using uncorrupted neighborhoods of the pixels for impulsive noise suppression with ANFIS, IEEE Trans. Image Process. (2007). doi:0.1109/TIP.2007.891067.
[15] G. Daoming, C. Jie, ANFIS for high-pressure water jet cleaning prediction, Surf. Coat. Technol. 201 (2006) 1629–1634.
[16] A. Depari, A. Flammini, D. Marioli, Andrea Taroni, Application of an ANFIS algorithm to sensor data processing, IEEE Trans. Instrum. Measur. 56 (1) (2007) 75–79.
[17] K. Assaleh, Extraction of fetal electrocardiogram using adaptive neuro-fuzzy inference systems, IEEE Trans. Biomed. Eng. 54 (1) (2007) 59–68.
[18] M.-L. Huang, H.-Y. Chen, J.-J. Huang, Glaucoma detection using adaptive neuro-fuzzy inference system, Expert Syst. Appl. 32 (2007) 458–468.
[19] A. Castro, V. Miranda, Mapping neural networks into rule sets and making their hidden knowledge explicit application to spatial load forecasting, in: Proceedings of the 14th Power System Computation Conference, 2002.

[20] S. Mitra, Y. Hayashi, Neuro-fuzzy rule generation: survey in soft computing framework, IEEE Transactions on Neural Networks 3 (2000) 748–768.
[21] J.-S.R. Jang, C.T. Sun, E. Mizutani, Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, Prentice-Hall, Englewood Cliffs, NJ, 1997.
[22] S. Effati, M. Jafarzadeh, A new nonlinear neural network for solving a class of constrained parametric optimization problems, Appl. Math. Comput. (2006), doi:10.1016/j.amc.2006.08.069.