Computational Intelligence and Automated Methods for Control Fuzzy System Design

Milan Todorovic
School of Engineering
RMIT University
Melbourne, Australia
milan.todorovic@rmit.edu.au

Milan Simic
School of Engineering
RMIT University
Melbourne, Australia
milan.simic@rmit.edu.au

***Abstract*—This paper aims to present a complex method of computational intelligence for a control fuzzy system design in a situation when there is no much of prior knowledge. Initial values are obtained by rule-based computations using available body of knowledge. Two fuzzy logic automated methods, batch and recursive least squares, are used to continue the computation and modelling of the system, to build and enhance the knowledge base. The extension principle methods are used in complex environments with both, discretized and continuous functions, with crisp and fuzzy data and transformations. The application of this model is illustrated by solving the autonomous cruise control problem, specifically, the throttle system. This computational method has an advantage comparing to artificial neural networks because the later do modeling of the system based only on learning data without knowing the nature of modelling applications.**

Keywords—Automated methods, Batch least squares, Recursive least squares, Control systems, Extension principle, Fuzzy systems.

I. Introduction

THIS paper presents an computational intelligence approach to design a control fuzzy system in a situation when there is no much of previous body knowledge. Considering this condition, the modelling starts with following the few provisionary rules, based on theoretical engineering knowledge.

The initial modelling relies on several assumptions. One of them is that it envisages the system as the closed-loop, i.e. feedback control system, where the physical system under control is regulated by the differential value between actual and desirable response. Fuzzy intervals are labelled with linguistic rules. Appropriate membership functions and relationships are established for states, input and output variables, which defuzzfied crisp results are used as inputs to physical controlling system. It can approximate the nonlinear function with predefined accuracy. As the alternative method could be used the artificial neural networks with their learning of system I/O data and using converging techniques up to the required performances but without knowing the nature of the problem.

If the body of knowledge is incomplete or too expensive to be comprehensive or linguistic rules are not reliable then new automated computational methods should be applied. Batch and recursive least square methods can be used when the body of knowledge is limited. They use outputs of the rule-based modelling as their initial values to continue with further modelling based on Gaussian membership functions.

The specific application which is used to illustrate this approach is one industrial problem in a domain of automotive industry. Specifically, it is related to the automatic cruise control of self-driving cars and the modelling of its throttle system depending on all variables that have impact on this system. In this example, one input variable is – speed and the second is a composite index of all other influence factors (road inclination, etc.). Some equations from the engineering body of knowledge and linguistic rule-base results are inputs for automated modelling methods which are also subsequently used to enhance the existing body of knowledge. This case and similar aspects of the transition from traditional to electrical and autonomous vehicles are more extensively discussed in [1], [2] and [3].

II. Control Fuzzy System Design

Control systems are one area where a paradigm shift has occurred after several cases where classical control methods were neither effective nor efficient what instigated the introduction of fuzzy control [4]. The basic model of controllable systems, assumes the existence of their inputs, outputs and control action parameters which are utilized to transfer the systems into required standing.

Regulatory control systems fulfil the purpose to uphold the certain physical parameters at some required level, despite the various instabilities while, in setpoint following controllers, these parameters should track some chosen function.

The most general representation of the control system can be modelled as the feedback system which output is attuned based on the error signal, which is a variance between the wanted reference input and the resulting output, gauged by the measurement system [5]. For a physical system, it is designed to modify, adjust or guide through the governing actions another physical system to present certain looked-for features or activities.

A fuzzy rule-based system uses a group of fuzzy provisional declarations raised from the pool of existing information to approximate, with a desirable accuracy, the nonlinear function which represents its control dynamics.

The artificial neural networks, for example, construct this nonlinear control function based on analogical learning from numerical input-output measurement data, using adjustive and congregating techniques and predefined operational conditions.

Some assumptions need to be made to achieve fuzzy control system feasibility: the plant need to satisfy conditions of observability and controllability. It must exist a learning discipline, containing a collection of language instructions or engineering guidelines, perceptions or a corpus of I/O quantitative data from which these instructions can be extricated. A ‘good-enough’ resolution of the problem, with satisfactory degree of accuracy, should be existent and viable, it does not need to be the ideal one. With issues of the solution solidity and optimality can be further dealt at the end of the design process.

A. Fuzzy logic control system

An example of fuzzy logic control system design can be presented as on the Fig. 1. The collection of language rules holds information about all I/O fuzzy partitions, respective membership functions, which determine input variables and control actions in a form of output variables to the controlling plant.

The design process of the fuzzy control system consists of the several stages. First need to be identified controller’s I/O-s and states. Intervals covered by each variable should be split into fuzzy subgroups which should be labelled with linguistic expression and their membership functions defined. The next is to establish fuzzy relationships amid their states, inputs and outputs, to normalize their variables and determine the basic rules. All inputs to the controller should be fuzzified and using previous rules and approximate reckoning didact, aggregate and defuzzify outputs to obtain crisp values. These values are inputs to the physical feedback system [6], [7].

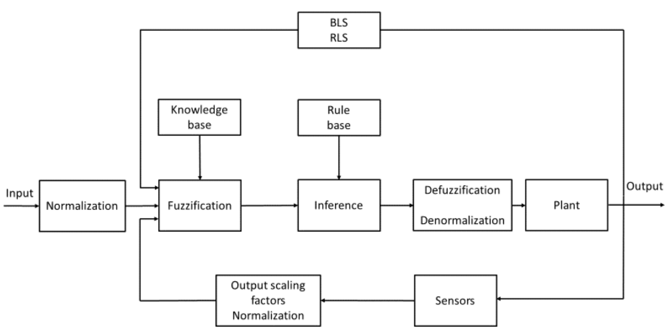

Fig. 1. Closed loop fuzzy logic control system (inspired by [8]).

III. The Problem – Throttle System

The following problem is from the domain of automotive industry. It is needed to simulate self-driving car’s cruise control system. The first step is to identify input variables (in this case – the speed and angle of inclination of the road) and output variables (the throttle position). The initial values are obtained by calculating system dynamics equations. After that it is needed to apply automated methods based on measured inputs and outputs [5].

The system dynamics is presented as:

$$t = c_1 s + \theta c_2 + ma \quad (1)$$

$$a = s(n+1) - s(n) \quad (2)$$

$$t(n) = c_1 s(n) + \theta(n) c_2 + m(s_{n+1} - s_n) \quad (3)$$

$$s_{n+1} = \left(1 - \frac{c_1}{m}\right) s(n) + t(n) - \frac{c_2}{m}\theta(n) \quad (4)$$

$$s_{n+1} = c_a s(n) + [1 - c_b] \begin{bmatrix} t(n) \\ \theta(n) \end{bmatrix} \quad (5)$$

Where,

t = throttle position $c_2 = mg \sin \Theta$
c_1 = friction coefficient a = acceleration
s = speed m = mass
Θ = angle of inclination $c_a = 1 - \frac{c_1}{m}$ and $c_b = \frac{c_2}{m}$

By choosing the convenient values for coefficients, c_1 / m and c_2 / m, and loosely deriving the previous equations for the illustrative purposes, we are getting:

$$\Theta_{k+1} = \Theta_k + 0.1 * V_k$$

$$V_{k+1} = V_k + T_k$$

Membership functions for state variables are presented in Table I.(A) – I.(B) and Fig. 2.A – 2.B. Membership functions for control output are presented in Table I.(C) and Fig. 2.C. The summarized rules are presented in FAM Table I.(C). FAM (fuzzy associative memory) table shows how the rule-based system conducts nonlinear mapping from the input to the output space of the fuzzy system. It starts from initial values and loosely derived differential control equations and state variables are updated for every cycle.

TABLE I
Membership Functions for State Variables and Control Output

(A) Θ-factor

	-5	-4	-3	-2	-1	0	1	2	3	4	5
Θ1	0	0	0	0	0	0	1/5	2/5	3/5	4/5	1
Θ2	0	0	0	0	1/5	2/5	3/5	4/5	1.0	4/5	3/5
Θ3	2/5	3/5	4/5	1.0	4/5	3/5	2/5	1/5	0	0	0
Θ4	1.0	4/5	3/5	2/5	1/5	0	0	0	0	0	0

(B) Speed

	0	10	20	30	40	50	60	70	80	90	100
V1	0	0	0	0	0	0	0	0	.5	1	1
V2	0	0	0	0	0	0	.5	1	.5	0	0
V3	0	0	0	0	.5	1	.5	0	0	0	0
V4	0	0	.5	1	.5	0	0	0	0	0	0
V5	1	1	.5	0	0	0	0	0	0	0	0

(C) Throttle

	0	1	2	3	4	5	6	7	8	9	10
T1	0	0	0	0	0	0	0	0	.5	1	1
T2	0	0	0	0	0	0	.5	1	.5	0	0
T3	0	0	0	0	.5	1	.5	0	0	0	0
T4	0	0	.5	1	.5	0	0	0	0	0	0
T5	1	1	.5	0	0	0	0	0	0	0	0

(D) FAM

	V1	V2	V3	V4	V5
Θ1	T3	T4	T5	T5	T5
Θ2	T2	T3	T4	T5	T5
Θ3	T1	T2	T3	T4	T5
Θ4	T1	T1	T3	T4	T4

For the first cycle, we have:

Initial speed: $v_0 = 10$;   Initial Θ-factor: $\theta_0 = -5$ degrees.

Throttle: to be computed

Θ-factor fires Θ3 at 2/5 and Θ4 at 1.0.

Speed fires only V5 at 1.0.

Θ-factor		Speed		Throttle
Θ3 (2/5)	AND	V5 (1.0)	→	T5(2/5)
Θ4 (1.0)	AND	V5 (1.0)	→	T4(1.0)

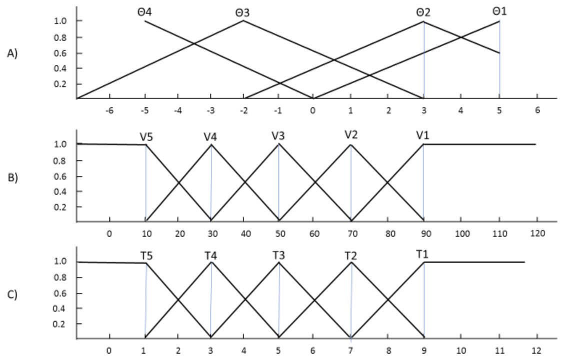

Fig. 2. Membership functions of state variables and control output.

Defuzzification, using centroid method, gives $T_0 = 3$. The result for cycle 1 appears in Fig. 3.A. Now we calculate new values of the state variables and throttle for the next cycle.

$\theta_1 = \theta_0 + 0.1 * v_0$ = - 5 + 1 = - 4

$v_1 = v_0 + T_0$ = 10 + 3 = 13

Θ-factor θ_1 = - 4 fires Θ3 at 3/5 and Θ4 at 4/5.

Speed v_1 = 13 fires V4 at 0.3 and V5 at 0.7.

Θ-factor		Speed		Throttle
Θ3(3/5)	AND	V1(0.3)	→	T4(0.3)
Θ3(3/5)	AND	V2(0.7)	→	T5(0.6)
Θ4(4/5)	AND	V1(0.3)	→	T3(0.3)
Θ4(4/5)	AND	V2(0.7)	→	T3(0.7)

Centroid gives T_1 = 2.98. Results are shown at Fig. 2.B.

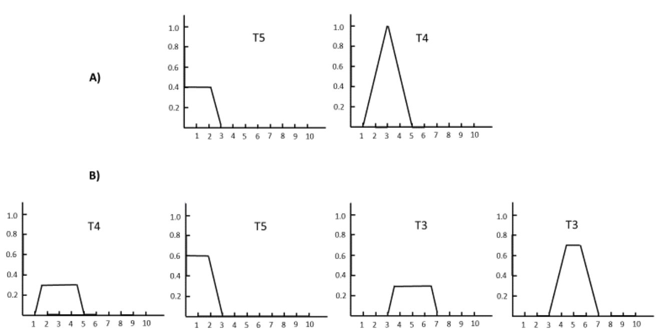

Fig. 3. Truncated sequences for cycles 2 and 3.

The summary of three cycles of simulation results are presented in Table II.

TABLE II
3-CYCLES SIMULATION RESULTS

	Cycle 0	Cycle 1	Cycle 2
Θ-factor	-5	-4	-2.7
Speed	10	13	16
Throttle	3	2.98	2.93

IV. BATCH AND RECURSIVE LEAST SQUARES

In many cases the available knowledge base is insufficient to precisely model of engineering systems and, because of this, the nonlinear mathematical approach is inadequate. Usually, the knowledge base is populated with measurement results from previous experiments. However, if the experiments are too expensive to conduct or we have emerging systems without much prior knowledge, it is difficult to model the controlling system based on incomplete conventional linguistic rules. That is the point when fuzzy modelling can step in to model the system with restricted and imperfect information. Some of these models which alleviate the lack of previous intelligence are batch least squares (BLS) and recursive least squares (RLS) automated methods, which enable the development of membership functions and enhancing the rules.

An example of 2-inputs, 1-output system is illustrated in (6) where the information is provided by three points and where the inputs are x_1 and x_2 and the output is y. For the inputs $\mu(x)$, this algorithm uses Gaussian membership functions.

$$\mu(x) = \exp\left[-\frac{1}{2}\left(\frac{x_i - a_i}{\sigma_i}\right)^2\right] \quad (6)$$

The output membership function is represented by a delta function. This is an impulse function which has a zero width and only one value, which full membership is positioned at b_i. It is sufficient for this algorithm which can consider any kind of output membership function [8].

Using Gaussian membership function for the input in this algorithm and delta functions as the output gives us the equation to foresee the output based on input data-tuple x_j:

$$f(x|\varphi) = \frac{\sum_{i=1}^{R} b_i \prod_{j=1}^{n} \exp\left[-\frac{1}{2}\left(\frac{x_j - a_j^i}{\sigma_j^i}\right)^2\right]}{\sum_{i=1}^{R} \prod_{j=1}^{n} \exp\left[-\frac{1}{2}\left(\frac{x_j - a_j^i}{\sigma_j^i}\right)^2\right]} \quad (7)$$

where, R and n are the numbers of rules and inputs per data-tuple. In our case, the system has two inputs x_1 and x_2, (i.e. n = 2), and two rules (i.e. R = 2). The letter φ is a vector with membership function parameters a_i, σ_i, b_i.

If we define the regression factor ψ_i (for i = 1, 2) as:

$$\psi_i(x) = \frac{\mu_{i\,(x)}}{\sum_{i=1}^{R} \mu_{i\,(x)}} = \frac{\prod_{j=1}^{n} \exp\left[-\frac{1}{2}\left(\frac{x_j - a_j^i}{\sigma_j^i}\right)^2\right]}{\sum_{i=1}^{R} \prod_{j=1}^{n} \exp\left[-\frac{1}{2}\left(\frac{x_j - a_j^i}{\sigma_j^i}\right)^2\right]} \quad (8)$$

We can use (9) to calculate the output. However, since we are using the BLS approach, the output is calculated differently [6]. The resulting mapping for BLS, approach is

$$f(x|\hat{\varphi}) = \hat{\varphi}^T \psi(x) \quad (9)$$

Where, $\hat{\varphi}$ is the least squares estimate vector from the training set, and $\hat{\varphi}^T$ is the transpose. The calculation of the least squares estimates and ψ_i are explained bellow.

For each input data-tuple, we have two (i = 2) values for ψ, one for Rule 1 and another for Rule 2, extracted from the Table II and resulting in a total of six values:

$\psi_1(x^1), \psi_2(x^1)$; $\psi_1(x^2), \psi_2(x^2)$; $\psi_1(x^3), \psi_2(x^3)$;

Using (8) we get:

$\psi_1(x^1) =$

$$\frac{\exp\left[-\frac{1}{2}\left(\frac{x_1-a_1^1}{\sigma_1^1}\right)^2\right]*\exp\left[-\frac{1}{2}\left(\frac{x_2-a_2^1}{\sigma_2^1}\right)^2\right]}{\exp\left[-\frac{1}{2}\left(\frac{x_1-a_1^1}{\sigma_1^1}\right)^2\right]*\exp\left[-\frac{1}{2}\left(\frac{x_2-a_2^1}{\sigma_2^1}\right)^2\right]+\exp\left[-\frac{1}{2}\left(\frac{x_1-a_1^2}{\sigma_1^2}\right)^2\right]*\exp\left[-\frac{1}{2}\left(\frac{x_2-a_2^2}{\sigma_2^2}\right)^2\right]}$$

and

$\psi_2(x^1) =$

$$\frac{\exp\left[-\frac{1}{2}\left(\frac{x_1-a_1^2}{\sigma_1^2}\right)^2\right]*\exp\left[-\frac{1}{2}\left(\frac{x_2-a_2^2}{\sigma_2^2}\right)^2\right]}{\exp\left[-\frac{1}{2}\left(\frac{x_1-a_1^1}{\sigma_1^1}\right)^2\right]*\exp\left[-\frac{1}{2}\left(\frac{x_2-a_2^1}{\sigma_2^1}\right)^2\right]+\exp\left[-\frac{1}{2}\left(\frac{x_1-a_1^2}{\sigma_1^2}\right)^2\right]*\exp\left[-\frac{1}{2}\left(\frac{x_2-a_2^2}{\sigma_2^2}\right)^2\right]}$$

For x^2 and x^3 of data set, we obtain the following values of $\xi_i(x)$:

$$\psi_1(x^1) = 0.679 \qquad \psi_2(x^1) = 0.321$$

$$\psi_1(x^2) = 0.148 \qquad \psi_2(x^2) = 0.852$$

$$\psi_1(x^3) = 0.130 \qquad \psi_2(x^3) = 0.987$$

With $\psi_i(x)$ completely specified, the transpose of $\psi_i(x)$ is determined and placed into a matrix, Θ:

$$\Theta = \begin{bmatrix}\psi^T(x^1)\\ \psi^T(x^2)\\ \psi^T(x^3)\end{bmatrix} = \begin{bmatrix}0.679 & 0.321\\ 0.148 & 0.852\\ 0.130 & 0.987\end{bmatrix} \qquad (10)$$

And, we have the following outputs placed in vector Y:

$$\mathrm{Y} = [\gamma^1 \quad \gamma^2 \quad \gamma^3] = [3 \quad 2.98 \quad 2.93] \qquad (11)$$

Using Y and Θ, we determine $\hat{\varphi}$,

$$\hat{\varphi} = (\Theta^T\Theta)^{-1}\Theta^T\mathrm{Y} \qquad (12)$$

Thus, producing φ

$$\hat{\varphi} = \begin{bmatrix}3.032\\ 2.946\end{bmatrix} \qquad (13)$$

Using (9), we calculate the output for the training data set:

$$f(x|\hat{\varphi}) = \hat{\varphi}^T\psi(x),$$

$$f(x^1|\hat{\varphi}) = 3.004$$

$$f(x^2|\hat{\varphi}) = 2.959$$

$$f(x^3|\hat{\varphi}) = 2.947$$

The fuzzy system maps the training data set reasonably accurately and the additional points, not in the training set, can be used as a test set to see how the system interpolates these new values.

The accuracy of the fuzzy model developed using BLS primarily depends on the rules specified in the rule-based and the data set used to train the fuzzy model.

RLS does not use all training data and does not compute inverse $\Theta^T\Theta$ each time when $\hat{\varphi}$ is updated. It calculates $\hat{\varphi}(k)$ at each time step k from the past estimate $\hat{\varphi}(k\text{-}1)$ and the latest data pair that is received x^k, γ^k.

For recursive procedure, we use a covariance to calculate $\hat{\varphi}$, which is computed using the regression vector and a covariant calculated at the previous step (16). To accomplish this, it needs to be first computed a preliminary covariance matrix $M(0)$ using a parameter beta (β) and the identity matrix I (14). $M(0)$ is the starting covariance matrix at the initial time period 0, (i.e. k = 0) and is used to compute the covariance matrices M, in the following time periods. Equation (15) establishes recursiveness, which enables the calculation of the M matrix for every time-period. The parameter beta should be greater than 0. It is chosen to be β = 2000 and I as the 2x2 identity matrix. For setting the initial values for $\hat{\varphi}$ at time-period 0 (k = 0) the best choice would be to use result obtained at batch procedure [6], therefore

$$\hat{\varphi}(0) = \begin{bmatrix}3.032\\ 2.946\end{bmatrix}$$

Instead of these values can be used any other collection of values may be used but more cycles may be needed to arrive at good values. As mentioned previously, this example demonstrates the only one cycle training of the fuzzy model.

$$M(k=0) = M(0) = \beta * I,$$

$$M(0) = 2000*\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} = \begin{bmatrix}2000 & 0\\ 0 & 2000\end{bmatrix} \qquad (14)$$

Once $M(0)$ is determined, we use it along with $\psi_i(x^{k=1})$ to calculate the next M and $\hat{\varphi}$ for the next step, M(k = 1) and $\hat{\varphi}(k=1)$. This is accomplished using (15) and (16):

$$M(k) = \frac{1}{\lambda}\{I - M(k-1)\psi_i(x^k)\left[\lambda I + (\psi_i(x^k))^T M(k-1)\,\psi_i(x^k)\right]^{-1}(\psi_i(x^k))^T\}M(k-1) \qquad (15)$$

$$\hat{\varphi}(k) = \hat{\varphi}(k-1) + M(k)\psi_i(x^k) * \left[y^k - (\psi_i(x^k))^T\hat{\varphi}(k-1)\right] \qquad (16)$$

For k = 1 and $\psi_1(x^1) = 0.679$, $\psi_2(x^1) = 0.321$

$$M(1) = \begin{bmatrix}366.30 & -771.70\\ -771.70 & 1635.47\end{bmatrix} \quad \text{and} \quad \hat{\varphi}(1) = \begin{bmatrix}-9.77\\ 4.55\end{bmatrix}$$

The computation process is repeated:

For k = 2 and $\psi_1(x^2) = 0.148$, $\psi_2(x^2) = 0.852$ and

For $k = 3$ and $\psi_1(x^3) = 0.130$, $\psi_2(x^3) = 0.987$

The vector parameters $\hat{\theta}$ is recursively calculated using (16) based on three inputs which are necessary to model the system. Having been determined it can be used, like the BLS case, in combination with ξ from (10) to calculate resultant outputs for the training data:

$$f(x|\hat{\varphi}) = \hat{\varphi}^T \psi(x) \quad (17)$$

V. Extension Principle

One of the problems related to the control fuzzy system design is the application of mathematical concepts to fuzzy numbers. The extension principle is a general approach to solve these problems. There are several ways how to translate extended fuzzy operations into effective computational procedures. The main approximation approach is to decompile membership functions into sequences of λ periods. The discretization of fuzzy numbers without adequate degree of resolution can bring unexpected results when extension principles are used [8].

If $\tilde{A}_1, \tilde{A}_2, \ldots \tilde{A}_n$ are input fuzzy sets defined on $X_1, X_2 \ldots X_n$ and the mappings of them are defined as $\tilde{B} = f(\tilde{A}_1, \tilde{A}_2, \ldots \tilde{A}_n)$, with a membership function:

$$\mu_{\tilde{B}}(y) = \max_{y=f(x1,x2,\ldots xn)} \{\min[\mu_{\tilde{A}_1}(x_1), \mu_{\tilde{A}_2}(x_2) \ldots \mu_{\tilde{A}_n}(x_n)]\} (18)$$

Equation (18), which expresses a discrete-valued function is called Zadeh's extension principle [9].

The *Vertex method* is another way to simplify applications of the extension principle for continuous-valued fuzzy variables. The method is based on a combination of the λ-cut concept and usual interval analysis. The Vertex method can prevent anomalies in the output membership function due to application of the discretization techniques on the fuzzy variables. The algorithm is easy to implement, and it can be computationally efficient [10].

The *DSW algorithm* also uses the λ-cut depiction of fuzzy sets. It consists of several steps. First, it is needed to select the value of λ, greater than 0 and lower than 1. Second, it should find the input membership functions intervals that relate to this λ. Third, the interval for the output membership function for the specific λ-cut can be computed. Previous steps should be repeated to finalize a λ-cut depiction of the resolution [11].

A nonlinear function $y = x + 2x^2 + x^3$ with a fuzzy input is used to illustrate the methodology. We decompose the membership function for the input into three λ-cut intervals for $\lambda = 0^+$, 0.5 and 1, to get intervals $I_{0^+} = [0.75, 3]$, $I_{0.5} = [1.375, 2.5]$ and $I_1 = [1, 1]$. For each λ-cut level, the functional mapping on the intervals gives:

$I_{0^+} = [0.75, 3]$

$B_{0^+} = [0.75, 3] + 2[0.75^2 + 3^2] + [0.75^3 + 3^3] = [0.75, 3] + 2[0.563, 9] + [0.422, 27] = [2.303, 48]$

$I_{0.5} = [1.375, 2.5]$

$B_{0.5^+} = [1.375, 2.5] + 2[1.375^2 + 2.5^2] + [1.375^3 + 2.5^3] = [1.375, 2.5] + [3.782, 12.50] + [2.600, 21.875] = [7.757, 24.375]$

$I_1 = [1, 1]$

$B_{1^+} = [1, 1] + 2[1^2 + 1^2] + [1^3 + 1^3] = [1, 1] + [2, 2] + [1, 1] = [4, 4] = 4$

The Fig. 4 (A), presents three intervals.

A. Example

In the following example we can compare the extension principle method with discretised membership functions with other two methods. In electric vehicle, the accelerator pedal is usually connected to a variable resistor or similar way for digital electronics to read changes in the pedal position. The driver accelerator request to the vehicle system controller is converted to electric motor power demand and sent to the motor controller. The motor controller then directs the motor to pull necessary power from the battery.

The current, $\tilde{I}$, impedance, $\tilde{R}$, and voltage, $\tilde{V}$, are supposed to be fuzzy variables. It is needed to find the arithmetic product for $\tilde{V} = \tilde{R} * \tilde{I}$, using the extension principles [12].

For two fuzzy sets from previous example, $\tilde{R}$ and $\tilde{I}$ with given membership functions, presented on Fig. 4 (B and C), we will compute $\tilde{U} = \tilde{R} * \tilde{I}$.

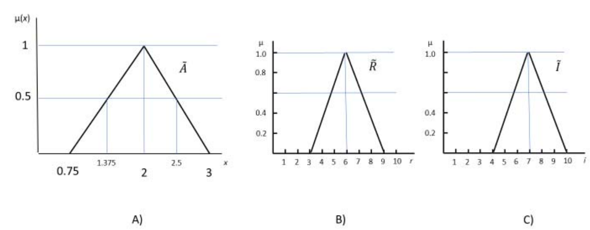

Fig. 4. Membership functions and -cuts intervals.

Extension method with discretised fuzzy sets and variables discretised in seven points:

$\tilde{R} = \left\{\frac{0}{3} + \frac{0.33}{4} + \frac{0.66}{5} + \frac{1.0}{6} + \frac{0.66}{7} + \frac{0.33}{8} + \frac{0}{9}\right\}$ and $\tilde{I} = \left\{\frac{0}{4} + \frac{0.33}{5} + \frac{0.66}{6} + \frac{1.0}{7} + \frac{0.66}{8} + \frac{0.33}{9} + \frac{0}{10}\right\}$

The voltage drop is:

$$\tilde{U} = \tilde{R}\ x\ \tilde{I} = \left\{\frac{0}{12} + \frac{0}{15} + \frac{0}{16} + \frac{0}{18} + \frac{0}{20} + \frac{0}{21} + \frac{0}{24} + \frac{0.33}{25} + \frac{0}{27} + \frac{0}{28} + \frac{0}{30} + \frac{0}{32} + \frac{0.33}{35} + \frac{0}{36} + \frac{0}{40} + \frac{0.66}{42} + \frac{0}{45} + \frac{0.33}{48} + \frac{0.66}{49} + \frac{0}{50} + \frac{0}{54} + \frac{0.33}{56} + \frac{0}{60} + \frac{0}{63} + \frac{0.33}{64} + \frac{0}{70} + \frac{0}{72} + \frac{0}{80} + \frac{0}{81} + \frac{0}{90}\right\}$$

Vertex method: I_{0^+}: Support (comprises those elements x such that $\mu_{\tilde{A}}(x) > 0$) for R is the interval [3, 9] and support for I is the interval [4, 10]

$r = 3, \quad i = 4 \quad\quad u = 12$

$r = 3, \quad i = 10 \quad\quad u = 30$

$r = 9, \quad i = 4 \quad\quad u = 36$

$r = 9, \quad i = 10 \quad\quad u = 90$

Thus, min = 12, max = 90, and $B_{0^+} = [12, 90]$.

$I_{0.33}$: R[4, 8], I[5, 9]

$r = 4, \quad i = 5 \qquad u = 20$

$r = 4, \quad i = 9 \qquad u = 36$

$r = 8, \quad i = 5 \qquad u = 40$

$r = 8, \quad i = 9 \qquad u = 72$

Thus, min = 20, max = 72, and B_{0^+} = [20, 72].

$I_{0.66}$: R[6, 10], I[6, 8]

$r = 6, \quad i = 6 \qquad u = 36$

$r = 6, \quad i = 8 \qquad u = 48$

$r = 10, \quad i = 6 \qquad u = 60$

$r = 10, \quad i = 8 \qquad u = 80$

Thus, min = 36, max = 80, and B_{0^+} = [36, 80].

$I_{1.00}$: R[6, 6], I[7, 7]

$r = 6, \quad i = 7 \quad u = 42$

Thus, min = 42, max = 42, and B_{0^+} = [42, 42].

DSW method:

I_{0^+}: [3, 9] ● [4, 10] = [min (12, 30, 36, 90), max (12, 30, 36, 90)] = [12, 90]

$I_{0.33}$: [4, 8] ● [5, 9] = [min (20, 36, 40, 72), max (20, 36, 40, 72)] = [20, 72]

$I_{0.66}$: [5, 7] ● [6, 8] = [min (30, 40, 42, 56), max (30, 40, 42, 56)] = [30, 56]

$I_{1.00}$: [6, 6] ● [7, 7] = [min (42, 42, 42, 42), max (42, 42, 42, 42)] = [42, 42]

VI. Conclusion

The objective of this paper is to present one method of computational intelligence for complex control fuzzy system design, but it can be used as a general approach for a range of different applications. It is comprised of several functional modules depending on the specific environment. It can begin with rule-based computations if sufficient body knowledge is available. In a case of emerging systems, it can use some methodologies to establish initial values and continue, as it is presented here, with batch and recursive least square methods. They can accomplish a desirable level of design accuracy and enhance the existing knowledge base. These methods can be supplemented by the extension principle or other means to address the fuzziness or crispness of the inputs, outputs and transformations, which can be both, discrete or continuous. Considering all these diverse aspects, including the inherent knowledge about the nature of the specific application, the significance of this approach is that it has some advantages in comparison with machine learning applications which rely mostly on large amount of the input-output data without having knowledge of the underlining design specifics.

References

[1] M. Todorovic and M. Simic, "Current State of the Transition to Electrical Vehicles". In Intelligent Interactive Multimedia Services. *Cham: Springer International Publishing*, 2019.

[2] M. Todorovic, M. Simic and A. Kumar, "Managing Transition to Electrical and Autonomous Vehicles". *Procedia Computer Science* 112, pp. 2335-2344, 2017.

[3] M. Todorovic and M. Simic, "Managing Transition to Autonomous Vehicles Using Bayesian Fuzzy Logic", *Intelligent Interactive Multimedia: Systems and Services*, KES2019, St. Julians, Malta. 17-19 June 2017.

[4] L. Zadeh, "A rationale for fuzzy control". *Journal of Dynamic System Measurement Transactions*, ASME, 94, pp. 3-4, 1972.

[5] C. L. Phillips and R. D. Harbor, *Feedback Control Systems*, 3rd edition, Prentice Hall, Englewood Cliffs, NJ, 1996.

[6] L. Zadeh, "The concept of linguistic variable and its application to approximate reasoning". Part I. *Information Science*, 8, pp. 199-249, 1975.

[7] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, New York, 1980.

[8] T. J. Ross, *Fuzzy Logic with Engineering Applications*, 3rd ed., Wiley, Washington, 2010.

[9] K. Passino and S. Yurkovich, *Fuzzy Control*, Addison-Wesley, Menlo Park, CA. 1998.

[10] R. R Yager, "A characterization of the extension principle". *Fuzzy Sets Systems,* 18, pp. 205-217, 1986.

[11] W. Dong and H. Shah, "Vertex method for computing functions of fuzzy variables", *Fuzzy Sets Systems*, 24, pp. 65-78, 1987.

[12] W Dong, H. Shah and F. Wong, "Fuzzy computations in risk and decision analysis". *Civil Engineering System*, 2, pp. 201-208, 1985.

[13] A. Kaufmann M. M. Gupta, *Introduction to Fuzzy Arithmetic – Theory and Applications*, Van Nostrand Reinhold, New York, 1985.