SISTEME DE REGLARE FUZZY

1. Introducere

În ultimii ani numărul şi variaţia aplicaţiilor logicii fuzzy a suferit un proces de largă expansiune. Gama aplicaţiilor porneşte de la produse de largă utilizare precum camere video, video–recordere, maşini de spălat şi cuptoare cu microunde până la reglarea proceselor industriale, instrumente medicale, sisteme de luare a deciziilor.

De ce se utilizează reglarea fuzzy? Unul din motivele orientării industriei spre acest domeniu este datorat concurenţei între companii (majoritatea japoneze) care utilizează sau au început să utilizeze reglarea fuzzy în produse competitive, avantajele utilizării acesteia fiind găsite în rapoartele economice de specialitate. Oricum discuţiile din literatură şi din forumurile de specialitate au dus la găsirea următoarelor motive:

1. Reglarea fuzzy este o tehnologie nouă şi în consecinţă dispar pretenţiile legate de patentul soluţiei pentru probleme tehnice similare.
2. In Japonia reglarea fuzzy este „cerută" de consumatori, din moment ce reprezintă o soluţie „high-tech". In acest caz, tehnicile fuzzy sunt utilizate din motive de piaţă (ca răspuns la cererea pieţei).
3. dezvoltarea de regulatoare fuzzy este mai uşor de învăţat şi necesită personal mai puţin pregătit decât în cazul regulatoarelor convenţionale. Ca rezultat producţia este mai ieftină.
4. Regulatoarele fuzzy sunt mai robuste decât cele convenţionale.
5. Regulatoarele fuzzy sunt mai adecvate pentru reglarea proceselor neliniare.

Ştiinţific, numai ultimele trei motive prezintă interes şi în cele ce urmează discuţia va fi focalizată în acest sens.

Regulatoarele fuzzy sunt reprezentate prin reguli **dacă-atunci** şi astfel furnizează o reprezentare a cunoştinţelor prietenoasă pentru utilizator. Reprezentarea poate fi văzută ca un limbaj foarte înalt de programare (very high level programming language), unde programul constă în reguli de tip if-then, iar compilatorul şi/sau interpretorul utilizează un algoritm de reglare neliniar. Astfel, programând cu ajutorul exprimărilor calitative, reprezentate prin reguli if-then, se obţine un program de lucru cu exprimări cantitative, furnizate de traductoare şi de elementele de execuţie. Acest proces va duce la pierderi de informaţie, pentru că nu există o unică translaţie de la o entitate calitativă la o reprezentare cantitativă, cu excepţia unor cazuri speciale. De exemplu, nu există o reprezentare unică pentru exprimarea calitativă „tensiune mare" într-o tensiune reală şi viceversa. Pentru că în reglare semnalul de ieşire de la regulator trebuie să fie precis din punct de vedere cantitativ (din moment ce fizic el reprezintă un semnal de acţionare a unui motor electric, hidraulic, a unei pompe etc.) sunt necesare tehnici speciale adiţionale pentru translatarea din domeniul calitativ în cel cantitativ.

Avantajul reglării fuzzy constă în posibilitatea „programării" cunoştinţelor operatorilor şi inginerilor de proces (formate din experienţa operării procesului) într-un mod uşor de înţeles şi prietenos.

Se spune de multe ori că reglarea fuzzy este mai robustă. Totuşi nu s-au găsit asemenea rezultate de cercetare care să demonstreze că regulatoarele fuzzy sunt mai robuste decât regulatoarele convenţionale în general. Un regulator fuzzy este în fapt o neliniaritate statică şi dacă acesta este mai robust decât un regulator convenţional depinde

de regulile ce definesc această neliniaritate statică. Astfel că „regulatoarele fuzzy sunt mai robuste” trebuie interpretat ca „regulatoarele fuzzy sunt mai robuste la modificări cunoscute ale parametrilor”. Cum se construieşte însă un regulator fuzzy mai robust rămâne o problemă din moment ce robustețea depinde de gradul de cunoaştere a procesului reglat.

Un motiv în utilizarea reglării fuzzy este că aceasta este mai adecvată în reglarea proceselor neliniare. Dacă un regulator fuzzy sau în genere unul neliniar este în principiu capabil să regleze un proces neliniar este o problemă ce depinde de intrările alese ale regulatorului. Se spune despre regulatoarele fuzzy că sunt superioare regulatoarelor convenționale în reglarea proceselor neliniare. Acest lucru este adevărat în măsura în care există cunoştințe adiționale despre neliniaritățile procesului.

Rezumând cele prezentate, putem afirma următoarele: reglarea fuzzy furnizează o metodă de a proiecta algoritmi de reglare într-o manieră prietenoasă şi furnizează capacitatea de a capta comportamentul neliniar de reglare uman care s-a dovedit a fi adecvat în multe probleme complexe de reglare. Cu o metodă de proiectare a regulatoarelor apropiată de gândirea şi percepția umană se reduce timpul de dezvoltare a aplicației şi este necesar un personal mai puțin pregătit în domeniul proiectării regulatoarelor. Beneficiul economic este evident. Trebuie reliefat faptul că, robustețea „regulatoarelor” umane este datorată în primul rând abilității oamenilor de a se adapta în medii în schimbare şi datorită capacității lor de învățare. In acest sens există cercetări în ultimul deceniu în domeniul reglării fuzzy adaptive.

Principalele arii de aplicație ale reglării fuzzy sunt următoarele:

1. Procese care pot fi reglate adecvat de oameni şi pentru care senzorii furnizează informații regulatorului fuzzy în mod similar cu informațiile utilizate de oameni în reglarea proceselor (exemple ale aplicațiilor logicii fuzzy sunt în conducerea automobilelor, maşini de spălat etc.). Astăzi sunt numeroase aplicații ale logicii fuzzy pentru produse casnice.
2. Procese care în mod curent sunt reglate cu algoritmi de reglare liniari şi necesită dezvoltări ulterioare în domeniul algoritmilor neliniari de reglare, cunoscute de operatori şi de inginerii de proces.

Un punct de vedere mai mult sau mai puțin critic asupra reglării fuzzy este dat de Elkan (1994), care susține că regulatoarele fuzzy au următoarele proprietăți:

- Regulatoarele fuzzy utilizează tipic mai puțin de 100 de reguli; deseori chiar mai puțin de 20 de reguli;
- Informația într-un regulator fuzzy este în mod uzual superficială, atât din punct de vedere static cât şi dinamic;
- Cunoştințele dintr-un regulator fuzzy reflectă în mod tipic corelațiile dintre intrările şi ieşirile regulatorului;
- Parametrii numerici ai unui regulator fuzzy sunt ajustați în timpul unui proces de învățare;
- Regulatoarele fuzzy utilizează operatori ai logicii fuzzy.

2. Legătura între reglarea fuzzy şi teoria sistemelor automate

Astăzi majoritatea algoritmilor de reglare sunt implementați pe calculator. Acest fapt implică măsurarea intrărilor în regulator la o anumită perioadă de eşantionare. De exemplu, clasicul algoritm de reglare PID liniar poate fi reprezentat astfel

$$c(t) = c_0 + k_p\left[e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right] \quad (1)$$

unde c(t) este comanda regulatorului spre proces, iar e(t) este semnalul de eroare dintre valoarea dorită şi cea măsurată a ieşirii procesului. O implementare pe computer a regulatorului PID poate fi reprezentarea incrementală (în fapt o ecuaţie cu diferenţe finite):

$$c_{PID}(k) = c_{PID}(k-1) + a_1 e(k) + a_2\Delta e(k) + a_3\Delta^2 e(k) \quad (2)$$

unde

$$\Delta e(k) = e(k) - e(k-1);$$

$$\Delta^2 e(k) = \Delta e(k) - \Delta e(k-1).$$

In cazul în care se consideră regulatoare PI sau PD apar următoarele ecuaţii cu diferenţe:

$$c_{PI}(k) = c_{PI}(k-1) + k_I e(k) + k_P\Delta e(k) \quad (3.a)$$

$$c_{PD}(k) = c_{PD}(k-1) + k_D\Delta e(k) \quad (3.b)$$

Ecuaţiile (2), (3.a) şi (3.b) pot fi comparate cu reprezentarea algebrică a unui hiperplan

$$y = a_0 + \sum_{i=1}^{n} a_i x_i\ . \quad (4)$$

O reprezentare schematică pentru relaţiile (3.a) şi (3.b) este prezentată în figura 1.

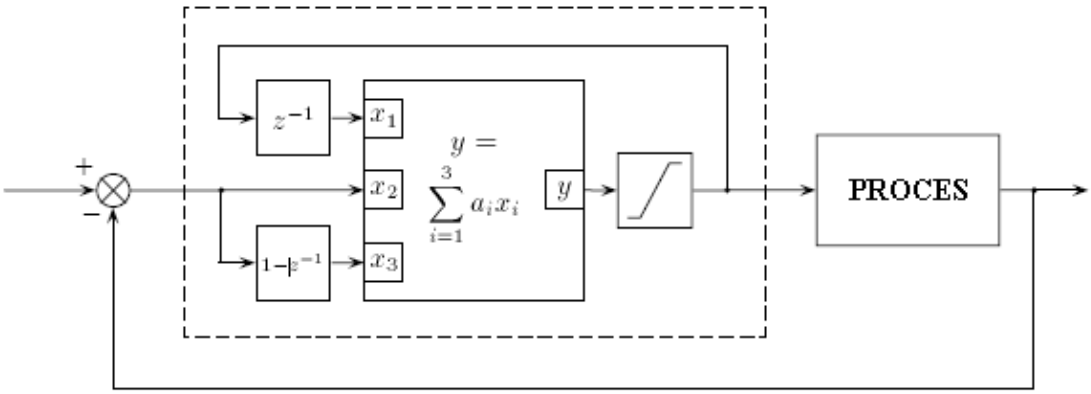


Fig.1. Exemplu de regulatoare PI şi PD privite ca o mapare statică utilizând blocuri pre şi post filtrare. Parametrii sunt aleşi astfel: $a_1 = 1$, $a_2 = k_I$ şi $a_3 = k_P$ pentru regulatorul PI, respectiv $a_1 = 0$, $a_2 = k_P$ şi $a_3 = k_D$ pentru regulatorul PD.

Când se consideră o mapare de la intrările regulatorului la ieşirile acestuia şi în general la sistemele MIMO funcţia regulatorului este reprezentată ca o mapare:

$$y = f(x).$$

Astfel, ieşirile regulatorului sunt considerate funcţii (mapări) statice ale intrărilor regulatorului. Comportamentul dinamic al regulatorului (cum ar fi acţiunea derivativă sau integrală) este emulat prin extinderea funcţiei regulatorului la mai multe intrări. Aceste intrări sunt întârzieri sau diferenţe ale unor intrări şi ieşiri. In acest mod, un regulator este considerat ca fiind alcătuit dintr-o funcţie statică a regulatorului şi părţi

adiționale de tipul prefiltrărilor sau post filtrărilor pentru obținerea semnalelor întârziate, intrări de tip diferență, integrări, limitări de semnal etc.

OBS: Această abordare este utilizată şi în general şi în domeniul rețelelor neuronale.

Cât priveşte stabilitatea sistemelor fuzzy trebuie remarcat faptul că regulatoarele fuzzy pot fi privite ca regulatoare neliniare şi din acest motiv este dificil de găsit un rezultat general în analiza şi sinteza regulatoarelor fuzzy (Driankov et al., 1993). Demonstrațiile de stabilitate pentru regulatoarele fuzzy găsite în literatură sunt restrânse la acele regulatoare fuzzy relativ simple (de ex. regulatoare fuzzy de tip PID) şi unde procesul de reglat este stabil. Dacă procesul nu poate fi modelat matematic (de exemplu cuptoarele cu ciment, Őstergaard, 1990) atunci nu pot fi date demonstrații de stabilitate.

Dezvoltarea teoriei moderne a reglării datorează mult modelării matematice, însă implementarea în viața reală a soluțiilor de reglare întâmpină de cele mai multe ori dificultăți datorate caracterului vag, imprecis al procesului reglat. Denumirea în engleză a acestui caracter este fuzzy. In prezent, termenul fuzzy se foloseşte cu valoare de adjectiv şi în limba română. Mulțimile fuzzy şi în general conceptele fuzzy au apărut din necesitatea de a exprima cantitativ „vagul", „imprecisul". Deşi există numeroase ramuri ale matematicii mai vechi decât teoria mulțimilor fuzzy, care se ocupă cu studiul proceselor de natură aleatoare: teoria probabilităților, statistica matematică, teoria informației şi altele, nu se pot face substituții între acestea şi teoria mulțimilor fuzzy.

3. Relația cu inteligența artificială

Logica fuzzy este privită ca una din tehnicile inteligenței artificiale din care fac parte şi sistemele expert „convenționale", rețelele neuronale şi algoritmii genetici. Zadeh (1994) propune denumirea de soft computing pentru domeniul rețelelor neuronale, algoritmilor genetici, logicii fuzzy şi combinațiilor acestora. Astăzi domeniul reglării şi modelării fuzzy este considerat în multe publicații că se suprapune celui corespunzător rețelelor neuronale. De altfel, în literatură se găsesc multe publicații despre sisteme neuro-fuzzy sau rețele neuro-fuzzy.

Un concept important în teoria mulțimilor fuzzy şi în logica fuzzy este variabila lingvistică (Zadeh, 1994). Legătura cu inteligența artificială apare şi din afirmația lui Dubois (1991) care spune că „motivul principal care a determinat apariția teoriei mulțimilor fuzzy este aparent dorința de a construi un cadru formal, cantitativ care să capteze vagul, imprecisul din cunoştințele umane aşa cum sunt ele exprimate prin limbaj natural". Din acest punct de vedere, logica fuzzy şi raționamentul aproximativ poate furniza un cadru pentru înțelegerea şi procesarea limbajului natural şi pentru modelarea felului în care oamenii raționează şi comunică.

Zadeh (1994) evidențiază diferența dintre reglarea fuzzy şi sistemele bazate pe cunoştințe după cum urmează: „Ceea ce realizează diferența între aplicațiile de reglare şi cele ale sistemelor bazate pe cunoştințe este că în reglare principala problemă ce trebuie rezolvată este imprecizia. Din contră în cazul sistemelor bazate pe cunoştințe problemele sunt legate atât de imprecizie cât şi de necunoscutele modelului". Reglarea fuzzy poate fi considerată o mică parte din cadrul teoretic al raționamentului aproximativ.

4. Mulțimi fuzzy (mulțimi vagi – fuzzy sets)

Noțiunea de mulțime vagă a fost introdusă în matematică şi teoria sistemelor de Zadeh (1965) sub denumirea de „mulțime fuzzy” care în traducere înseamnă mulțime neclară, estompată şi care se foloseşte sensul de vag, imprecis. In prezent termenul fuzzy se foloseşte cu valoare de adjectiv şi în limba română. Mulțimile fuzzy şi în general conceptele fuzzy au apărut din necesitatea de a exprima cantitativ „vagul”, „imprecisul”.

Teoria mulțimilor clasice este binecunoscută, iar în teoria mulțimilor fuzzy este denumită în mod obişnuit ca „teoria clasică a mulțimilor” în loc de „teoria mulțimilor”. Funcția de apartenență $\mu_A(x)$ a elementului x la o mulțime clasică A, ca submulțime a universului X, este definită prin:

$$\mu_A(x) = \begin{cases} 1 & \text{dacă } x \in A \\ 0 & \text{dacă } x \notin A \end{cases} \tag{5}$$

Aceasta înseamnă că un element x este fie membru al mulțimii A ($\mu_A(x) = 1$) sau nu ($\mu_A(x) = 0$). Mulțimile clasice mai sunt denumite şi mulțimi rigide, ferme, tranşante (crisp sets). Insă în multe clasificări nu este clar dacă elementul x aparține sau nu mulțimii A. De exemplu, dacă mulțimea A este alcătuită din PC-urile care sunt prea scumpe pentru bugetul unui student, atunci este evident că mulțimea nu are margini clar definite. Desigur se poate spune că un PC de 2500$ este prea scump, dar ce se poate spune de un PC la 2495$ sau 2502$? Sunt acestea prea scumpe sau nu? Se poate stabili însă o limită peste care un PC este prea scump pentru un student (să zicem 2500$) şi o limită sub care un PC nu este prea scump, să zicem 1000$. Intre aceste limite, rămâne totuşi un interval în care nu se poate preciza dacă un PC este prea scump sau nu. In acest interval poate fi utilizat un grad care să clasifice prețul ca parțial prea scump. Aici apare rolul mulțimilor fuzzy: mulțimi ale căror funcții de apartenență au grade de apartenență în intervalul [0,1].

O mulțime fuzzy (concept introdus de Zadeh, 1965) este o mulțime caracterizată de grade de apartenență în intervalul real $\mu_A(x) \in [0,1]$. O mulțime fuzzy A, o submulțime fuzzy a universului X este descrisă prin:

$$A = \sum_{i=1}^{m} \mu_A(x_i)/x_i = \mu_A(x_1)/x_1 + \ldots + \mu_A(x_m)/x_m \tag{6}$$

unde $\mu_A(x)$ este funcția de apartenență, X este cunoscut ca universul de discuție. Când universul X nu este finit, o mulțime fuzzy A este definită ca:

$$A = \int_X \mu_A(x)/x. \tag{7}$$

De exemplu, în afirmația „umiditatea este mare” , umiditatea este universul de discuție în timp ce „mare” reprezintă o mulțime fuzzy.

Pentru exemplul anterior, mulțimea fuzzy reprezintă acele PC-uri prea scumpe pentru bugetul unui student. Universul de discuție este reprezentat de PC-uri cu prețuri variate. Mulțimea fuzzy A este descrisă în figura 2.

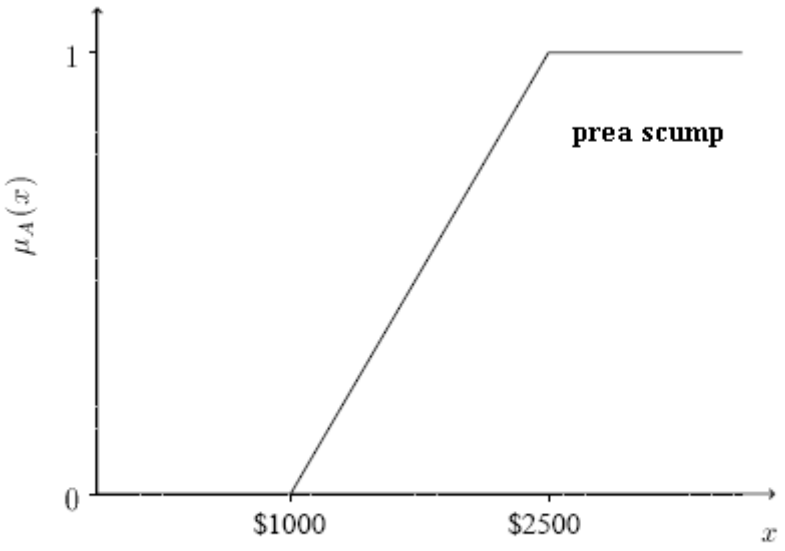


Fig.2. Mulțimea fuzzy A reprezentând PC-uri prea scumpe pentru bugetul unui student. Variabila x reprezintă prețul.

Din figura 2 se poate observa că dacă prețul este sub 1000$ în mod sigur PC nu este prea scump şi dacă prețul este peste 2500$, PC-ul este clasificat ca prea scump. Intre cele două limite poate fi observat un grad de apartenență crescător cu prețul. Nu este neapărat necesar ca funcția de apartenență să fie crescătoare sau discontinuă pentru 1000$ sau 2500$. Alegerea funcției de apartenență a unei mulțimi fuzzy este arbitrară.

4.1. Proprietăți ale mulțimilor fuzzy

In cele ce urmează s-a urmărit prezentarea celor mai utilizate proprietăți ale mulțimilor fuzzy şi nu o tratare exhaustivă a acestora.

Înălțimea unei mulțimi fuzzy A („height”) $hgt(A)$ este definită ca:

$$hgt(A) = \sup_{x \in X} \mu_A(x) \tag{8}$$

iar mulțimile fuzzy cu înălțimea egală cu 1 sunt denumite normale. Mulțimile fuzzy sunt subnormale dacă $hgt(A) < 1$. Nucleul unei mulțimi fuzzy (găsit în literatură drept kernel, nucleus, core) este o submulțime fermă a lui X:

$$core(A) = \{x \in X \mid \mu_A(x) = 1\} \tag{9}$$

Suportul unei mulțimi fuzzy este tot o mulțime fermă a lui X:

$$s(A) = \{x \in X \mid \mu_A(x) > 0\} \tag{10}$$

Dacă suportul unui set fuzzy este definit atunci se mai numeşte suport compact. Figura 3 prezintă noțiunile de nucleu, înălțime şi suport pentru o mulțime fuzzy.

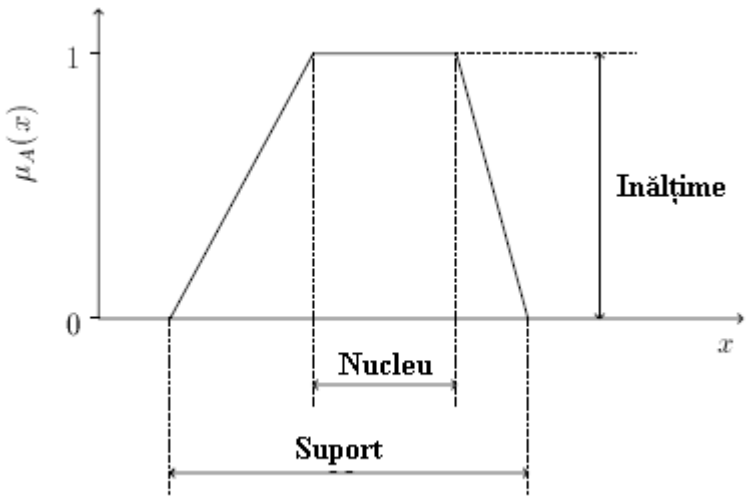


Fig.3 Înălțime, suport şi nucleu pentru mulțimea fuzzy A.

Elementele pentru care $\mu_A(x) = \frac{1}{2}$ sunt denumite puncte de trecere („crossover points”).

O tăietură α a unei mulțimi fuzzy este definită ca:

$$\alpha - cut(A) = \{x \in X \mid \mu_A(x) \geq \alpha\}. \quad (11)$$

O tăietură α a unei mulțimi este denumită în literatura de specialitate mulțime nivel. O tăietură α strictă este definită prin

$$\bar{\alpha} - cut(A) = \{x \in X \mid \mu_A(x) > \alpha\} \quad (12)$$

Astfel, nucleul unei mulțimi fuzzy poate fi definit ca o tăietură α cu α=1

$$core(A) = 1 - cut(A)$$

iar suportul poate fi definit ca o tăietură α strictă, cu α=0

$$s(A) = \bar{0} - cut(A).$$

O mulțime fuzzy convexă este caracterizată prin

$$\forall x_1, x_2, x_3 \in X, \quad x_1 \leq x_2 \leq x_3 \rightarrow \mu_A(x_2) \geq \min(\mu_A(x_1), \mu_A(x_3)) \quad (13)$$

unde x_1, x_2 şi x_3 sunt valori ale lui X. Astfel, mulțimea fuzzy din figura 3 este convexă.

Convexitatea unei mulțimi fuzzy poate juca un rol important în combinație cu o altă proprietate importantă a mulțimilor fuzzy care apare atunci când mulțimile fuzzy formează o partiție fuzzy. Când N_A mulțimi fuzzy A_j sunt submulțimi ale universului X, un asemenea tuplu de mulțimi fuzzy $(A_1,..., A_j,...A_{N_A})$ se numeşte partiție fuzzy dacă:

$$\forall x \in X, \quad \sum_{j=1}^{N_A} \mu_{A_j}(x) = 1. \quad (14)$$

ştiind că $A_j \neq \Phi$ şi $A_j \neq X$.

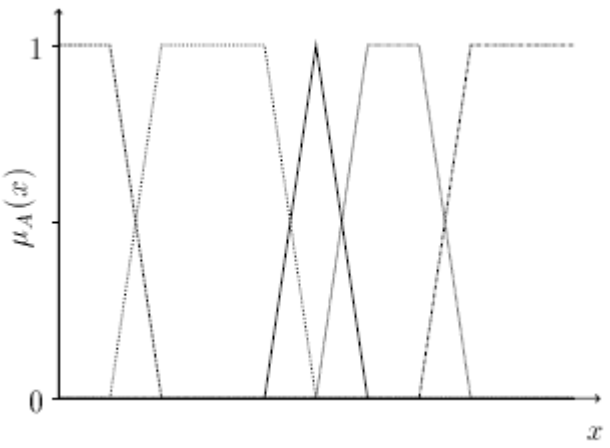


Fig. 4. Exemplu de partiție fuzzy.

Un exemplu de partiție fuzzy este dată în figura 4. O partiție fuzzy formată din mulțimi fuzzy normale şi convexe nu poate conține mai mult de două mulțimi fuzzy suprapuse.

4.2. Numere şi intervale fuzzy

Un număr fuzzy este un tip special de mulțime fuzzy. O mulțime fuzzy F este un număr fuzzy, de obicei o submulțime fuzzy a lui R, dacă îndeplineşte următoarele criterii:

- Mulțimea fuzzy este convexă, aşa cum s-a definit prin (13);
- Mulțimea fuzzy este normalizată: $hgt(F) = 1$;
- Funcția de apartenență a mulțimii fuzzy este continuă pe interval;
- Nucleul mulțimii fuzzy constă într-o singură valoare.

Astfel, un număr fuzzy este întotdeauna o mulțime fuzzy însă nu şi viceversa. Un exemplu al numărului fuzzy „în jur de 5” este prezentat în fig.5a. Operațiile matematice de adunare, scădere etc. pot fi extinse numerelor fuzzy cu ajutorul principiului extensiei.

In afară de numere fuzzy, pot fi considerate şi intervale fuzzy (Dubois şi Prade, 1988). Un interval fuzzy este o mulțime fuzzy cu aceleaşi restricții ca cele definite pentru numerele fuzzy excepție făcând nucleul care poate avea mai multe valori. Un exemplu de interval fuzzy „de la în jur de 2 la în jur de 7” este prezentat în fig.5b.

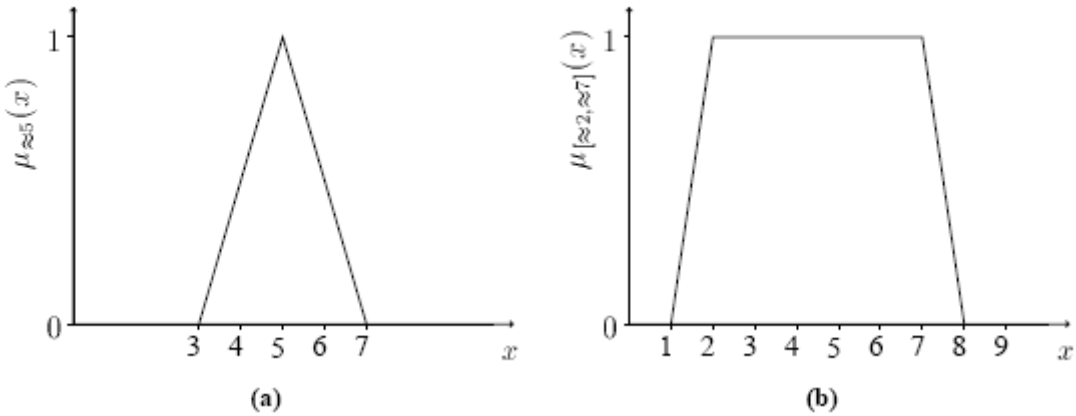


Fig.5. Mulțimi fuzzy reprezentând numărul fuzzy „în jur de 5” şi intervalul fuzzy „de la în jur de 2 la în jur de 7”.

4.3. Principiul extensiei

Principiul extensiei a fost introdus de Zadeh (1975) şi este unul din cele mai importante elemente ale teoriei mulțimilor fuzzy. Este principiul care extinde concepte matematice non-fuzzy mulțimilor fuzzy, scopul fiind exprimarea cantitativă a acestora. Principiul extensiei permite extinderea unei mapări f de la puncte din X la submulțimi fuzzy ale lui X:

$$f(A) = f(\mu_1 / x_1 + ... + \mu_n / x_n) = \frac{\mu_1}{f(x_1)} + ... + \frac{\mu_n}{f(x_n)} \qquad (15)$$

De exemplu, pentru numărul anterior „în jur de 5” cu un univers discret, $x_i \in \mathrm{Z}$ şi mapând funcția f reprezentând pătratul numărului fuzzy, principiul extensiei permite

$$(\text{„în jur de 5”})^2 = \left(\frac{\frac{1}{2}}{4} + \frac{1}{5} + \frac{\frac{1}{2}}{6} \right)^2 = \frac{\frac{1}{2}}{16} + \frac{1}{25} + \frac{\frac{1}{2}}{36}$$

Principiul extensiei aplicat unei funcții sau operații matematice $f(x_1,...,x_n)$, …, x_n) este definit prin:

$$\mu_B = \sup_{\substack{x_1,...,x_n \\ y=f(x_1,...,x_n)}} \mu_A(x_1,...,x_n) = \sup_{\substack{x_1,...,x_n \\ y=f(x_1,...,x_n)}} \min(\mu_{A_1}(x_1),...,\mu_{A_n}(x_n))$$

unde produsul cartezian $A_1 \times ... \times A_n$ este utilizat pentru reprezentarea mulțimii fuzzy A multidimensionale.

Ex. Când universurile sunt discrete, $x_i \in \mathrm{Z}$, adunarea a două numere fuzzy de exemplu „în jur de 2” cu „în jur de 5” se reduce la

$$\text{„în jur de 2”+„în jur de 5”} = \left(\frac{\frac{1}{2}}{4} + \frac{1}{5} + \frac{\frac{1}{2}}{6} \right) + \left(\frac{1}{2} \right) = \left(\frac{\frac{1}{2}}{6} + \frac{1}{7} + \frac{\frac{1}{2}}{8} \right) \text{ (vezi fig.6).}$$

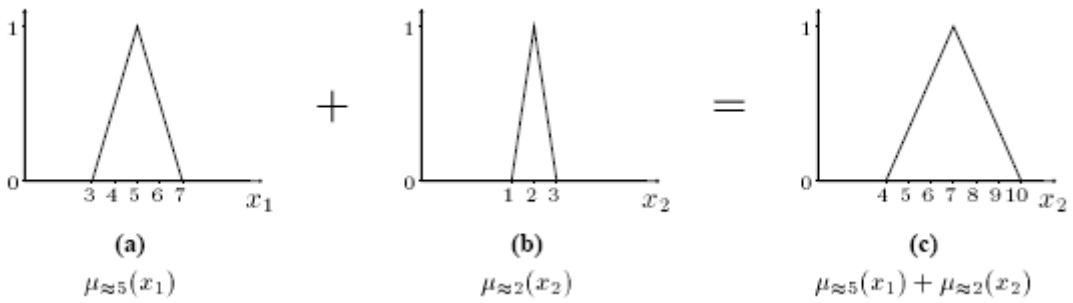


Fig.6. Rezultatul adunării a două numere fuzzy.

4.4. Reprezentări ale mulțimilor fuzzy

Majoritatea operațiilor cu mulțimi sunt definite pentru universuri continue, iar operațiile sunt definite şi pentru universuri discrete pentru cazuri speciale. In literatura de specialitate pot fi distinse trei tipuri de reprezentări:

1. *Reprezentarea funcțională*. Acest tip de reprezentare a unei mulțimi fuzzy utilizează descrieri funcționale pentru reprezentarea mulțimilor fuzzy:

$$\mu_A(x) = f(x) \tag{16}$$

Un exemplu este descrierea funcțională a mulțimii fuzzy cu funcție de apartenență triunghiulară (fig.7):

$$f(x;a,b,c) = \begin{cases} 0 & x \le a \\ \dfrac{x-a}{b-a} & a \le x \le b \\ \dfrac{c-x}{c-b} & b \le x \le c \\ 0 & c \le x \end{cases}$$

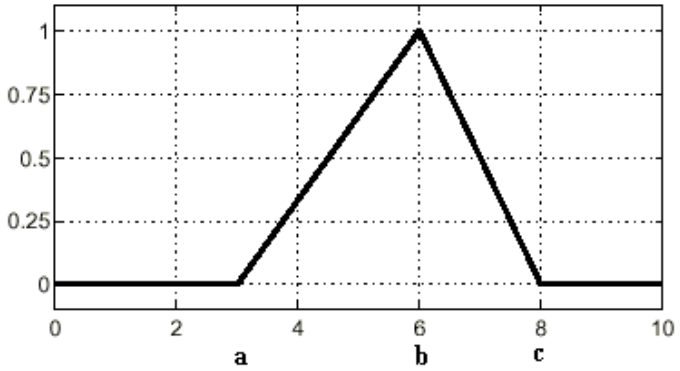


Fig.7. Funcția de apartenență triunghiulară.

2. *Reprezentarea pereche.* Acest tip de reprezentare a unei mulțimi fuzzy defineşte o mulțime fuzzy ca

$$\mu_A(x) = \frac{\mu_1}{x_1} + \frac{\mu_2}{x_{21}} + ... + \frac{\mu_n}{x_n} \tag{17}$$

Această reprezentare este naturală în cazul mulțimilor fuzzy pe domenii discrete. De exemplu, să considerăm mulțimea fuzzy „prietenii lui Viorel”, care este reprezentată de o mulțime de persoane identificate cu numele lor (x_i) şi de gradul de apartenență (μ_i) ataşate reprezentând gradul de prietenie.

3. Reprezentarea în nivel. Aceasta descrie o mulțime fuzzy prin tăieturile sale (α-cuts)*:*

$$\mu_A(x) = \sup_{[0,1]} \alpha - cut(\mu_A(x)) \tag{18}$$

Din cele prezentate până acum rezultă că reprezentarea pereche este alegerea evidentă pentru implementarea mulțimilor fuzzy pe computer. Pentru domeniile continue reprezentarea funcțională este utilă doar în cazurile când operațiile sunt simple.

5. Modificatori lingvistici

Modificatorii lingvistici pot fi utilizați pentru modificarea înțelesului unei mulțimi fuzzy. De exemplu, modificatorul lingvistic *foarte* poate fi utilizat pentru schimbarea înțelesului lui mare în foarte mare. Unii autori au denumit aceşti modificatori lingvistici

bariere (hedges). Există în acest sens două abordări: modificatori gradați (powered hedges) şi modificatori de mutare (shifted hedges), modificatorii scalați (scaled hedges) fiind o combinație a avantajelor celor două abordări.

5.1. Modificatori gradați

Modificatorii lingvistici operează asupra gradelor de apartenență şi sunt reprezentate de

$$m_p(A) = \int_X \mu_A^p(x)/x \qquad (19)$$

unde m_p este modificatorul şi p este parametrul specific unui anumit modificator lingvistic. Sunt alese valori implicite pentru p, de exemplu 2 pentru *foarte* (concentrare), ½ pentru *mai mult sau mai puțin* (dilatarea – „oarecum") etc. De exemplu: $foarte(A) = \int_X \mu_A^2(x)/x$, fig.7a.

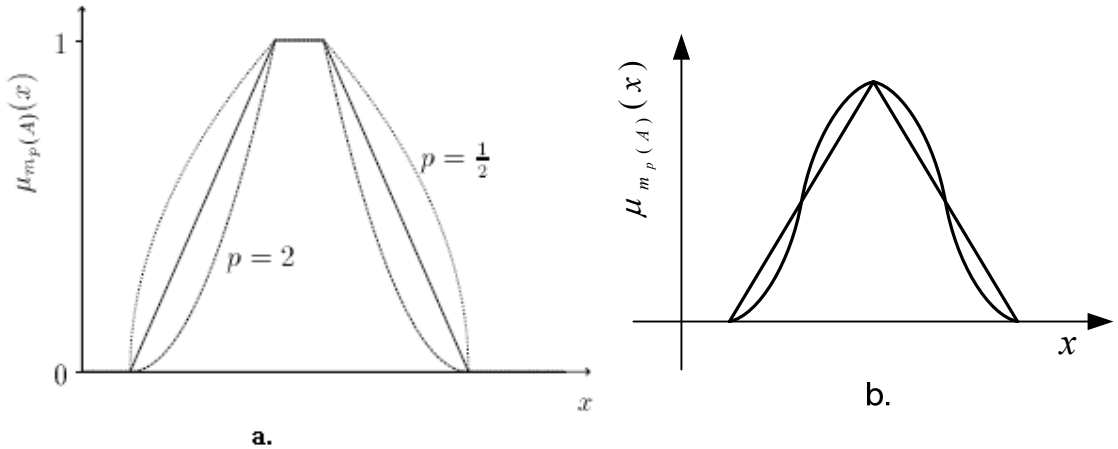


Fig.7. Mulțimea fuzzy A şi modificatorii lingvistici asociați.

In fig. 7a este dat un exemplu de modificatori lingvistici foarte şi mai mult sau mai puțin. Avantajul acestor modificatori gradați este că pentru fiecare modificator o operație standard poate fi definită prin alegerea unei valori standard a lui p. Pot fi desprinse următoarele proprietăți ale lui $m_p(A)$ pentru diferite valori ale lui p:

$0 < p < 1$; mulțimea fuzzy se dilată $m_p(A) \supset A$;

$p = 1$; mulțimea fuzzy nu se modifică $m_p(A) = A$;

$p > 1$; mulțimea fuzzy se concentrează $m_p(A) \subset A$.

Suportul şi nucleul unei mulțimi fuzzy nu se modifică prin aplicarea modificatorilor gradați.

Intensificarea („într-adevăr") are ca efect mărirea gradelor de apartenență de peste 0,5 şi micşorarea celor sub 0,5 (fig.7b):

$$\mu_{\text{int}(A)} = \begin{cases} 2(\mu_A(x))^2, \mu_A(x) \le 0{,}5 \\ 1 - 2(1-\mu_A(x))^2, \mu_A(x) > 0{,}5 \end{cases}.$$

5.2. Modificatori de mutare (shifted hedges)

Modificatorii de mutare sunt definiți de relația

$$m_s(A) = \int_X \mu_A(x-s)/x \qquad (20)$$

unde m_s este modificatorul lingvistic, iar s reprezintă mărimea mutării şi ia mai multe valori în cadrul unui modificator lingvistic. Când se aplică un asemenea modificator lingvistic (de exemplu *foarte*) unei mulțimi fuzzy de formă trapezoidală, valoarea lui s va fi pozitivă şi în partea stângă şi negativă în partea dreaptă a centrului unei mulțimi fuzzy. In acest fel mulțimea fuzzy se concentrează rezultând $m_s(A) \supset A$.

In fig.8. este prezentat efectul unor asemenea modificatori.

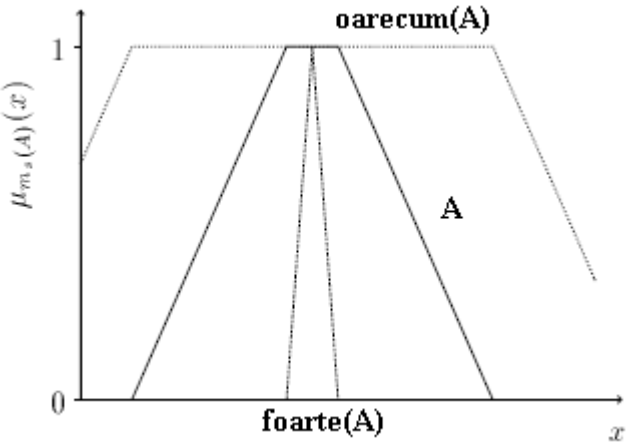


Fig.8 Mulțimea fuzzy A şi foarte(A), mai mult sau mai puțin(A).

In cazul modificatorului *foarte*, metoda constă în crearea unui suport pentru $m_s(A)$ de mărimea nucleului lui A şi reducerea nucleului cu aceeaşi cantitate dacă este posibil. Pentru funcții de apartenență triunghiulare această reducere a nucleului nu este posibilă. Pentru modificatorul lingvistic *mai mult sau mai puțin (oarecum)* este aplicată o metodă complementară (din moment ce este considerat modificatorul lingvistic complementar lui *foarte*).

5.3. Modificatori scalați (scaled hedges)

Aceştia combină avantajele modificatorilor gradați (o operație standard pe o mulțime fuzzy poate fi definită pe un operator specific) şi pe cele ale modificatorilor de mutare (forma mulțimii fuzzy este conținută în cea rezultată după modificare).

$$m_c(A) = \int_X \mu(c(x-r_A)+r_A)/x = \int_X \mu(cx+(1-c)r_A)/x \qquad (21)$$

unde

c - factor de scalare al modificatorul m_c;

r_A - punctul de referință pentru mulțimea fuzzy A.

Factorul de scalare c este standard pentru modificatori lingvistici specifici. Pot fi alese aceleaşi valori ca pentru modificatorii gradați; de exemplu, pentru modificatorul lingvistic foarte ($c = 2$) se poate scrie:

$$foarte(A) = \int_X \mu(2(x-r_A)+r_A)/x = \int_X \mu(2x-r_A)/x.$$

Punctul de referință r_A este un punct caracteristic mulțimii fuzzy A. In cazul funcțiilor de apartenență convexe, valoarea punctului de referință poate fi aleasă în centrul nucleului funcției de apartenență. In fig.9 sunt prezentate rezultatele modificatorilor lingvistici *foarte* şi *mai mult sau mai puțin*. Aşa cum se poate observa, modificatorii păstrează forma originală a funcției de apartenență de bază.

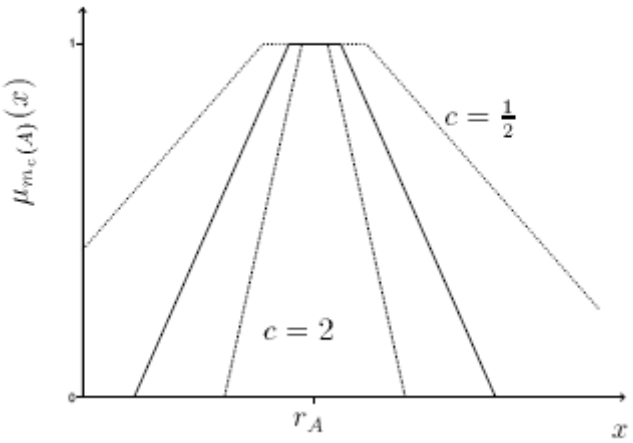


Fig.9 Modificatori scalați.

6. Operații cu mulțimi fuzzy

Aşa cum sunt definiți operatorii pentru mulțimile clasice, operatori similari sunt definiți şi pentru mulțimile fuzzy. Intersecția, reuniunea sau complementul unei mulțimi sunt operații cunoscute de la teoria mulțimilor clasice. Operatorii în acest caz sunt definiți în mod unic şi sunt prezentați în tabelul 6.1.

Tabelul 6.1

A	B	A∩B	A∪B	Not(A)
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Aceste operații sunt definite şi în teoria mulțimilor fuzzy. Totuşi, pentru că gradul de apartenență poate lua valori în intervalul [0,1] aceşti operatori nu pot fi definiți în mod unic.

6.1. Reuniunea şi intersecția

Extinderea intersecției şi reuniunii a două mulțimi clasice la intersecția şi reuniunea a două mulțimi fuzzy nu este definită în mod unic.

Zadeh a propus utilizarea următoarelor definiții:

$\mu_{A\cap B} = \min(\mu_A(x), \mu_B(x))$ intersecție

$\mu_{A\cup B} = \max(\mu_A(x), \mu_B(x))$ reuniune.

Dacă $\mu_A(x)$ şi $\mu_B(x)$ iau valori doar în $\{0,1\}$, aceşti operatori se reduc la operatorii intersecție şi reuniune definiți pentru mulțimile clasice. Totuşi avându-se în vedere intervalul de valori [0,1] în care $\mu_A(x)$ şi $\mu_B(x)$ pot lua valori, rezultă o infinitate de soluții posibile de implementare a intersecției şi reuniunii. Forme generale pentru intersecție şi reuniune sunt reprezentate de norme triunghiulare (T-norms) şi conorme triunghiulare (T-conorms sau S-norms).

O normă T este o funcție de două variabile definită pe [0,1]x[0,1] cu valori în [0,1] care satisface criteriile:

1. $T(a,1) = a$
2. $T(a,b) \le T(c,d)$ dacă $a \le c, b \le d$
3. $T(a,b) = T(b,a)$
4. $T(T(a,b),c) = T(a,T(b,c))$

Normele triunghiulare satisfac restricția:

$$T_w(a,b) \le T(a,b) \le \min(a,b) \tag{22}$$

unde $T_w(a,b)$ este norma T triunghiulară propusă de Weber, cunoscută şi sub numele de produs drastic, definită prin

$$T_w(a,b) = \begin{cases} a & \text{if b} = 1 \\ \text{b} & \text{if a} = 1 \\ 0 & \text{in rest} \end{cases} \tag{23}$$

Suprafețele inferioare şi superioare pentru o normă T generală sunt prezentate în fig. 10, a şi c.

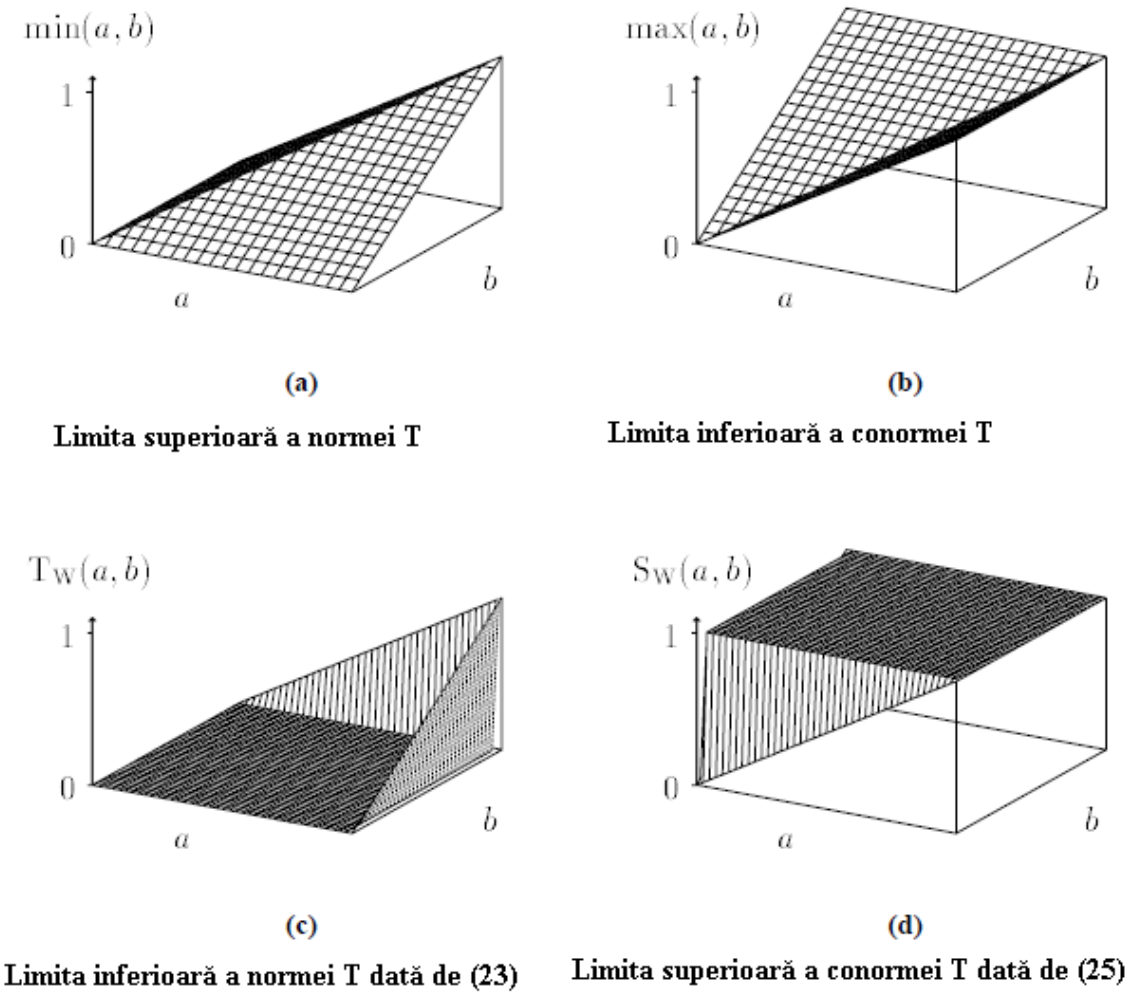


Fig. 10 Limitele asociate normelor T şi conormelor T.

Condiţiile pentru definirea unei conorme T sunt în afară de 2,3 şi 4

S.1 $S(a,0) = a$

O conormă T satisface restricţia

$$\max(a,b) \le S(a,b) \le S_w(a,b) \qquad (24)$$

unde S_w este norma S (conorma T) propusă de Weber (1983) cunoscută drept „sumă drastică” şi definită ca:

$$S_w(a,b) = \begin{cases} a & \text{if b} = 0 \\ \text{b} & \text{if a} = 0 \\ 1 & \text{in rest} \end{cases} \qquad (25)$$

şi este prezentată în fig. 10 d. Fig. 10 b prezintă limita inferioară pentru o normă S generală (operatorul max).

6.2 Complementul unei mulţimi fuzzy

Complementul $\bar{A}$ al unei mulţimi fuzzy A se defineşte prin:

C-1. $c(0)=1$

C-2. $c(a)<c(b)$ dacă $a>b$

C-3. $c(c(a))=1$

Zadeh a sugerat definiția $c(a)=1-a$. In literatura de specialitate se găsesc foarte multe definiții pentru complementul unei mulțimi fuzzy, una din cele mai utilizate definiții fiind cea a lui Sugeno şi anume λ-complementul unei mulțimi fuzzy A:

$$\mu_{\bar{A}^{\lambda}}(x)=\frac{1-\mu_A(x)}{1+\lambda\mu_A(x)}, \text{ cu } \lambda>0 .$$

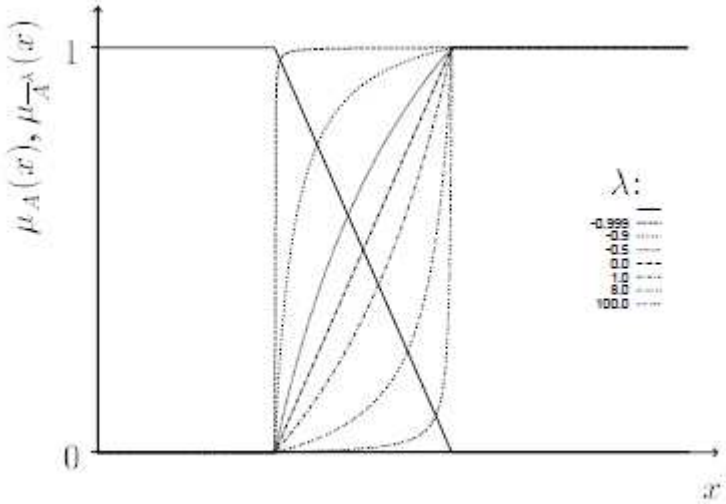


Fig.11. λ-complementul unei mulțimi fuzzy.

In figura 11 sunt date câteva exemple de λ-complementul unei mulțimi fuzzy pentru diverse valori ale lui λ.

7. Relații fuzzy

Până acum funcțiile de apartenență au fost doar unidimensionale. Mulțimile fuzzy pot avea funcții de apartenență multidimensionale aceste multidimensionale sunt referite în literatura de specialitate ca relații fuzzy.

O relație fuzzy R (n-dimensională) în $X_1\times\ldots\times X_n$ este o submulțime (multi-dimensională) fuzzy a lui $X_1\times\ldots\times X_n$ şi se notează ca:

$$R=\{\mu_R(x_1,\ldots,x_n)/(x_1,\ldots,x_n)\mid x_1\in X_1,\ldots,x_1\in X_n\} \qquad (26)$$

In logica fuzzy, relațiile fuzzy sunt utilizate pentru modelarea asociațiilor lingvistice de tipul *mai mic decât*, *cam de 2 ori mai în vârstă*, *mult mai ieftin* etc.

De exemplu, relația fuzzy „x aproximativ egal cu y” notată cu $R_{\approx}(x,y)$ este reprezentată în figura 12.

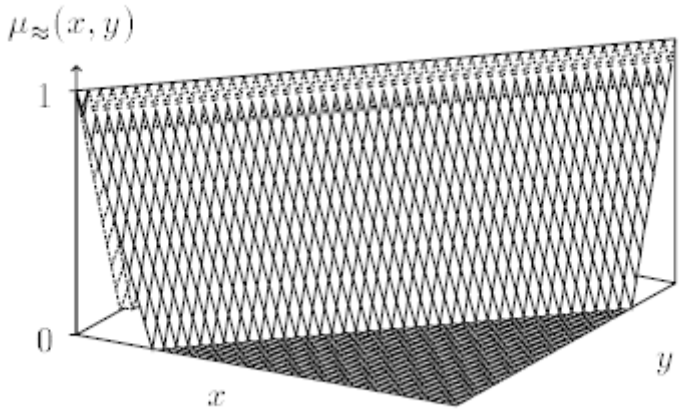


Fig.12. Reprezentarea relației x aproximativ egal cu y.

In forma discretă, utilizând valori întregi în intervalul $[0,10]$ relația fuzzy se descrie cu:

$$R_{\approx} = 1/(0,0) + \frac{1}{2}\Big/(1,0) + \frac{1}{2}\Big/(0,1) + 1/(1,1) + \frac{1}{2}\Big/(2,1) + \frac{1}{2}\Big/(1,2) + 1/(2,2) + \frac{1}{2}\Big/(3,2) +$$
$$+ \ldots + \frac{1}{2}\Big/(10,9) + \frac{1}{2}\Big/(9,10) + 1/(10,10)$$

In forma continuă, presupunând că x şi y sunt numere reale, relația fuzzy din figura 12 va fi:

$$R_{\approx} = \int_{X \times Y} \mu_{R_{\approx}}(x,y)/(x,y)$$

unde funcția de apartenență este:

$$\mu_{R\approx}(x,y) = \max(1 - \frac{1}{2}|x-y|, 0)/(x,y).$$

Zadeh a definit proiecția unei relații fuzzy R în $X^i = X_{i_1} \times \ldots \times X_{i_k}$ prin:

$$proj(R; X^i) = \int_{X^i} \sup_{x_{j1},\ldots,x_{jl}} \mu_R(x_{i1},\ldots,x_{ik})/(x_{i1},\ldots,x_{ik})$$

unde R este o submulțime fuzzy a lui $X^n = X_1 \times \ldots \times X_n$ şi $X^i \times X^j = X^n$. Indicii $j_1,\ldots,j_l$ sunt complementari lui $i_1,\ldots,i_k$ în concordanță cu indicii $1,\ldots,n$. Definiția proiecției poate părea complicată, însă rezultatul ei destul de simplu se vede în figura 13. Se proiectează o relație definită pe $X \times Y$ pe o mulțime fuzzy definită pe Y:

$$B = proj(R;Y) = \int_Y \sup_y \mu_R(x,y)/y.$$

In afara mecanismului proiecției, Zadeh a introdus şi conceptul complementar de extensie cilindrică a unei relații (sau mulțimi) fuzzy:

$$cext(R; X^n) = \int_{X^n} \mu_R(x_{i1},\ldots,x_{ik})/(x_{i1},\ldots,x_{ik}),$$

unde R este o relație fuzzy definită pe X^i. In figura 14 este dat un exemplu de mulțime fuzzy pe universul X şi extensia ei pe $X \times Y$.

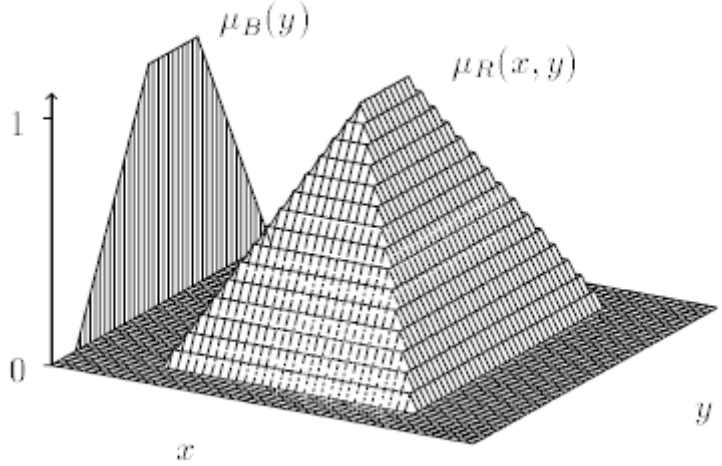


Fig.13. Rezultatul proiecției B=proj(R;Y).

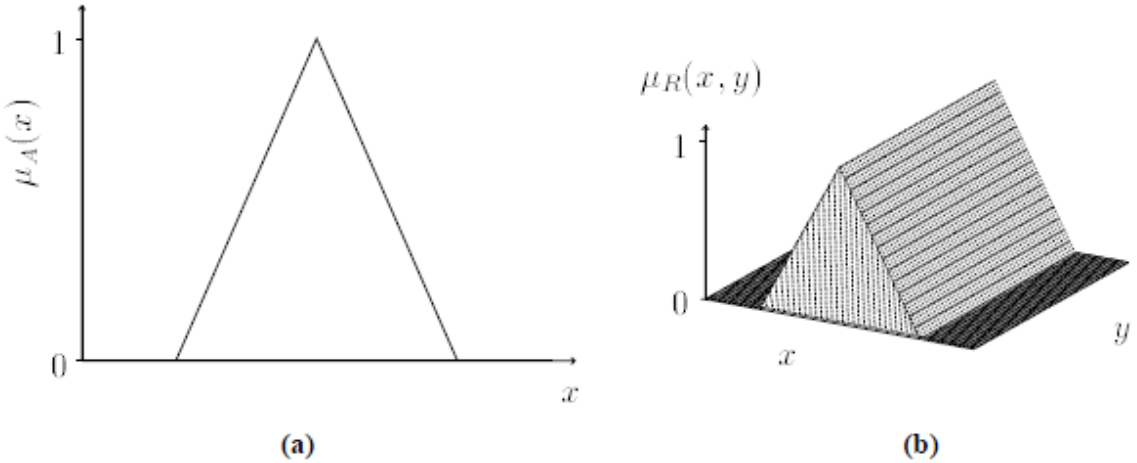


Fig.14. Rezultatul (b) al proiecției cilindrice a mulțimii fuzzy A (a).

Compunerea relațiilor fuzzy este definită de Zadeh (1973): dacă există o relație fuzzy R definită pe $X \times Y$ şi A este o mulțime fuzzy pe X, atunci submulțimea B pe universul Y poate fi indusă de A prin compunerea lui R şi A:

$B = A \circ R$

şi este definită de

$B = proj(R \cap cext(A; X \times Y); Y)$.

Zadeh a propus o compunere de tip sup-min,

$\mu_B(y) = \sup_x \min(\mu_A(x), \mu_R(x,y))$

Compunerea relațiilor fuzzy este alcătuită din două faze: combinarea şi proiecția. Compunerea relațiilor este un concept foarte important pentru logica şi raționamentul fuzzy (următorul capitol).

8. Logica şi raţionamentul fuzzy

Aşa cum teoria mulţimilor clasice stă la baza logicii clasice, teoria mulţimilor fuzzy stă la baza logicii fuzzy. Operaţiile cu mulţimi ca reuniunea, intersecţia şi complementul unei mulţimi au drept corespondent logic pe *sau*, *şi* şi *negaţie*. In cazul mulţimilor fuzzy, operatorii logici vor avea însă mai multe reprezentări faţă de cazul mulţimilor clasice unde reprezentările erau unice.

8.1. Propoziţii fuzzy

Un concept important în logica fuzzy îl reprezintă propoziţia fuzzy. Propoziţiile fuzzy reprezintă declaraţii de tipul „x este mare” unde „mare” este eticheta lingvistică, definită printr-o mulţime fuzzy pe universul de discuţie al variabilei x. Etichetele (lingvistice) fuzzy mai sunt cunoscute şi sub numele de constante fuzzy, termeni fuzzy sau noţiuni fuzzy. Propoziţiile fuzzy realizează legătura între variabile şi etichetele de lingvistice definite pentru aceste variabile.

Propoziţiile fuzzy reprezintă baza pentru logica şi raţionamentul fuzzy. Ele pot fi combinate prin intermediul conectorilor logici de tipul *şi* şi *sau*. Modificatorii lingvistici pot fi utilizaţi pentru modificarea înţelesului etichetei lingvistice utilizate într-o propoziţie fuzzy. De exemplu, modificatorul lingvistic foarte poate fi utilizat pentru schimbarea de la „x este mare” în „x este foarte mare”.

8.1.1 Conectori logici

Ca şi în cazul mulţimilor clasice, propoziţiile fuzzy pot fi combinate utilizând conectorii logici *şi* şi *sau*. Conectorii logici *şi* şi *sau* sunt implementaţi cu norme T respectiv conorme T. Aşa cum s-a văzut în cap. 6 există un număr infinit de norme T şi conorme T. Nu există recomandări generale de alegere a unei anumite norme sau conorme T într-o situaţie specifică. Pentru acelaşi univers de discuţie, în fig.15 sunt date exemple de rezultate ale aplicării normelor şi conormelor propuse de Łukasiewicz comparate cu cele propuse de Zadeh.

Operatorii propuşi de Zadeh au avantajul ignorării redundanţei, în sensul că, combinând două propoziţii fuzzy egale, rezultatul combinaţiei va reprezenta aceeaşi informaţie adică:

$$\begin{aligned} \mu_{A\cap A} &= \min(\mu_A(x), \mu_A(x)) = \mu_A(x) \\ \mu_{A\cup A} &= \max(\mu_A(x), \mu_A(x)) = \mu_A(x) \end{aligned} \tag{27}$$

Cei mai utilizaţi operatori în logica fuzzy sunt cei din tabelul 8.1.1

Tabelul 8.1.1

şi	sau	obs
Min(a,b)	Max(a,b)	Zadeh
Max(a+b-1,0)	Min(a+b,1)	Łukasiewicz
ab	a+b-ab	probabilitate

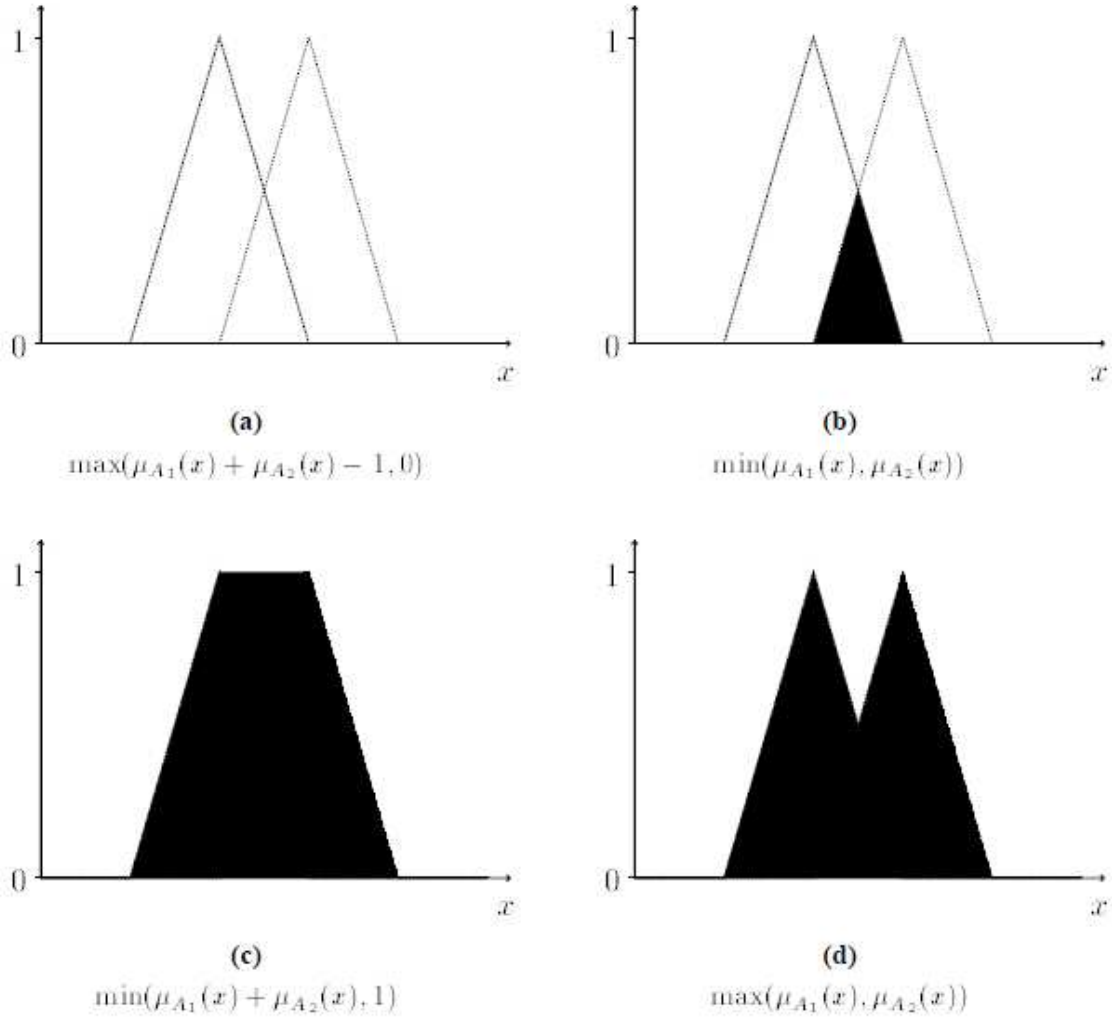


Fig.15. Rezultatele normelor T(a) şi conormelor T(c) propuse de Łukasiewicz şi ale normelor T(b) şi conormelor T(d) propuse de Zadeh pentru conectorii logici şi şi sau. Obs. Rezultatul în fig. a este 0.

Aşa cum se vede din fig.15 c conectorul *sau* aplicat pentru două propoziții fuzzy duce la o mulțime fuzzy cu un grad de apartenență complet între valorile aflate „între" mulțimile fuzzy (suma a+b fiind chiar mai mare ca 1). Dacă se consideră două propoziții fuzzy „nivelul de zgomot este ridicat" şi „nivelul de zgomot este coborât", în mod intuitiv rezultatul obținut prin aplicarea operatorilor propuşi de Łukasiewicz este mai potrivit. In general, alegerea normelor T şi a conormelor T depinde de înțelesul conexiunii logice şi de contextul în care aceasta e făcută.

Dacă propozițiile se referă la universuri diferite, aplicarea conectorilor logici va avea ca rezultat o relație fuzzy. De exemplu, să considerăm următoarea propoziție fuzzy:

p: x_1 este A_1 **şi** x_2 este A_2

unde A_1 şi A_2 au funcțiile de apartenență $\mu_{A_1}(x_1)$ şi $\mu_{A_2}(x_2)$. Propoziția p poate fi reprezentată de relația fuzzy P cu funcția de apartenență:

$$\mu_P(x_1, x_2) = T\left(\mu_{A_1}(x_1), \mu_{A_2}(x_2)\right) \tag{28}$$

unde T este o normă T generală care este folosită să modeleze conectorul *şi*. O asemenea combinație de propoziții, în fapt o propoziție fuzzy poate reprezenta premisa unei reguli fuzzy.

8.1.2. Negația în propozițiile fuzzy

Similar cu conectorii logici *şi* şi *sau* care se referă la intersecția şi reuniunea mulțimilor fuzzy, negația în propozițiile fuzzy se referă la complementul unei mulțimi fuzzy. Un exemplu simplu de propoziție fuzzy cu negație este

„Nivelul de zgomot *nu* este ridicat”

şi utilizând complementul fuzzy standard $c(a) = 1 - a$ rezultă:

$$\mu_{not\ ridicat}(x) = 1 - \mu_{ridicat}(x) \ .$$

In general, negarea într-o propoziție fuzzy „x *nu* este A” reprezintă

$$not(A) = \int_X c(\mu_A(x))/x \qquad (29)$$

unde complementul unei mulțimi verifică criteriile C-1, C-2 şi C-3 de la paragraful 6.2. In mod uzual este utilizat complementul standard:

$$\mu_{not(A)}(x) = 1 - \mu_A(x). \qquad (30)$$

8.2. Reguli fuzzy

Regulile fuzzy sunt reprezentate ca o funcție de implicație. O asemenea implicație fuzzy are aceeaşi funcție ca tabelul de adevăr al implicației clasice din logica clasică (tabelul 8.2). In logica clasică, implicația se notează cu

$$A \to B \qquad (31)$$

care poate fi văzută ca o reprezentare a declarației

dacă A ***atunci*** B.

In logica fuzzy aceste tipuri de declarații sunt deseori referite ca declarații fuzzy dacă-atunci sau reguli fuzzy.

Tabelul 8.2

A	B	Not(A)	Not(A)UB	$A \to B$
0	0	1	1	1
0	1	1	1	1
1	0	0	0	0
1	1	0	1	1

8.2.1. Reprezentarea unei reguli fuzzy

O regulă fuzzy este o declarație *dacă–atunci* unde premisa şi consecința sunt propoziții fuzzy. Premisa poate conține o combinație de propoziții realizată prin intermediul conectorilor logici *şi* şi *sau*. Este posibil ca premisa să conțină şi negații. Astfel, regula fuzzy

Dacă x_1 este A_1 ***şi Dacă*** x_2 este A_2 ***atunci*** y este B

are mulțimile fuzzy A_1, A_2 şi B reprezentate de funcțiile de apartenență $\mu_{A_1}(x_1)$, $\mu_{A_2}(x_2)$ şi $\mu_B(y)$. Pentru această regulă poate fi construită următoarea relație R:

$$R = I(T(A_1, A_2), B) \qquad (32)$$

unde T este o conjuncție bazată pe o normă T generală şi I este funcția de implicație fuzzy. Aşa cum norma T reprezintă (modelează) conectorul *şi*, funcția de implicație fuzzy I reprezintă (modelează) implicația, conectorul *dacă–atunci.* Astfel o regulă fuzzy poate fi reprezentată ca o relație fuzzy. Funcția de apartenență a lui R din exemplul anterior este dată de:

$$\mu_R(x_1, x_2, y) = I\left(T\left(\mu_{A_1}(x_1), \mu_{A_2}(x_2)\right), \mu_B(y)\right) \tag{33}$$

Funcțiile de implicație fuzzy sunt descrise în secțiunile următoare. Funcția de implicație I este notată $I(a,b)$ unde a,b$\in$[0,1].

8.2.2. Implicații fuzzy

Aşa cum orice operator din logica clasică are o infinitate de reprezentări în logica fuzzy, implicația fuzzy respectă şi ea această regulă. In afara implicațiilor fuzzy care respectă înțelesul implicațiilor clasice (ca în tabelul 8.2), implicația este implementată uneori ca o conjuncție, caz în care relația de cauzalitate dictată de expresia *dacă–atunci* nu este păstrată de relația fuzzy care reprezintă implicația.

Dubois şi Prade (1991) au realizat o clasificare a diferitelor tipuri de implicații fuzzy după cum urmează:

- Implicații în logica fuzzy bazate pe logica clasică; o fuzificare a implicației materiale (unde $a \to b$ este definit ca $\bar{a} \vee b$)

$$I(a,b) = S(c(a), b) \tag{34}$$

care mai sunt denumite şi implicații S.

- Implicații fuzzy bazate pe implicația din logica quantum (QL)

$$I(a,b) = S(c(a), T(a,b)) \tag{35}$$

care mai sunt denumite şi implicații QL. Lee (1990) a denumit acest tip de implicație calcul propozițional şi a definit calculul şi calculul propozițional extins

$$I(a,b) = S(T(c(a), c(b)), b) \tag{36}$$

care rezultă din (35) în care se înlocuiesc a şi b cu $1-a$ şi $1-b$.

- Implicații fuzzy care reflectă ordinea parțială a propozițiilor

$$I(a,b) = \begin{cases} 1 & \text{daca} \quad \text{a} \le \text{b} \\ 0 & \text{daca} \quad \text{a} = 1 \quad \text{si b} = 0 \\ \in [0,1) & \text{in rest} \end{cases} \tag{37}$$

aceste implicații sunt denumite şi implicații R.

- Implicații interpretate ca o conjuncție

$$I(a,b) = T(a,b) \tag{38}$$

unde T este o normă T. Acest tip de implicații T este utilizat în reglarea fuzzy (vezi cap. 9).

Există multe alte tipuri de implicații care nu aparțin categoriilor menționate, exemple în acest sens fiind prezentate în fig.16.

8.2.3. Agregarea regulilor fuzzy

Combinarea regulilor fuzzy într-o relație fuzzy reprezintă agregarea regulilor fuzzy. Se consideră că se dau N_r reguli paralele care au premisa bazată pe N_x variabile:

r_1 : *dacă* x_1 este $A_{1,1}$ *şi* …*şi* x_{N_x} este $A_{N_x,1}$ *atunci* y este B_1
altfel
…
altfel
r_k : *dacă* x_1 este $A_{1,k}$ *şi* …*şi* x_{N_x} este $A_{N_x,k}$ *atunci* y este B_k
altfel
…

altfel

r_{N_r} : *dacă* x_1 este A_{1,N_r} *şi* …*şi* x_{N_x} este A_{N_x,N_r} *atunci* y este B_{N_r} .

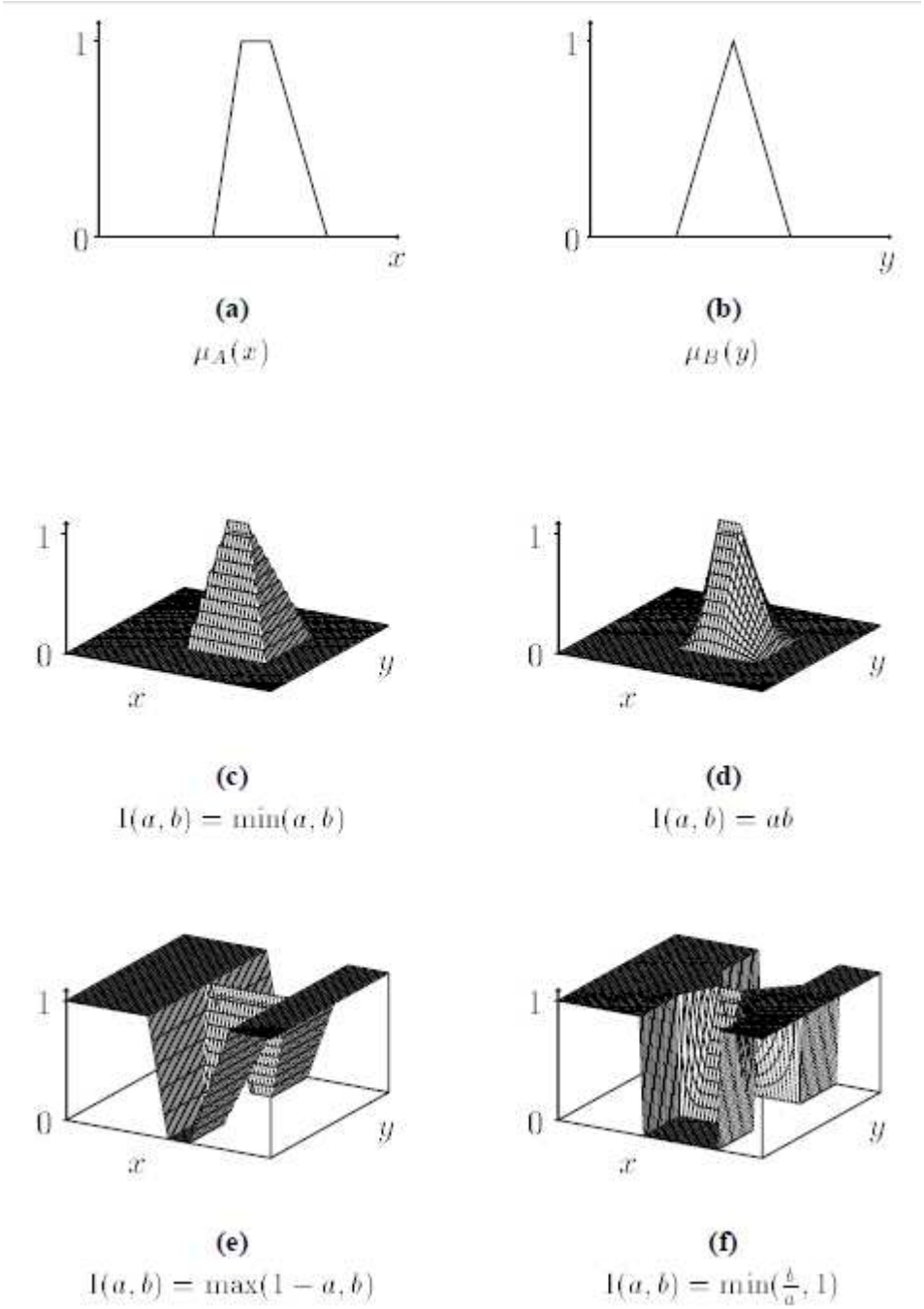


Fig.16. Relațiile fuzzy $R = I(a,b)$ ca rezultat al funcțiilor de implicație propuse de Mamdani (c), Larsen (d), Kleene-Davis (e) şi Goguen (f). Primele două figuri prezintă mulțimile fuzzy utilizate în premisa (a) şi consecința (b) regulii fuzzy: *dacă* x este A *atunci* y este B.

Traducerea unui asemenea set de reguli fuzzy într-o relație fuzzy este realizată prin construirea unei relații fuzzy R_k pentru fiecare regula r_k şi combinarea acestor relații într-o singură relație R. Această combinare a regulilor fuzzy într-o relație fuzzy se numeşte agregare. Iar această agregare se face diferit în funcție e tipul implicației fuzzy folosite.

$R = \bigcup_k R_k$ pentru implicațiile care se supun conjuncției clasice şi $R = \bigcap_k R_k$ pentru implicațiile care se supun implicației clasice. Pentru implicațiile care se supun conjuncției clasice operatorul de agregare este disjuncția. Pentru implicațiile care se supun implicației clasice operatorul de agregare este conjuncția. In fig.17 (f) este prezentată agregarea a două reguli fuzzy utilizând operatorul *max*. Operatorul min este utilizat ca funcție de implicație pentru cele două reguli din figură.

Inferența este o procedură care permite determinarea de noi cunoştințe pornind de la datele concrete ale problemei de rezolvat.

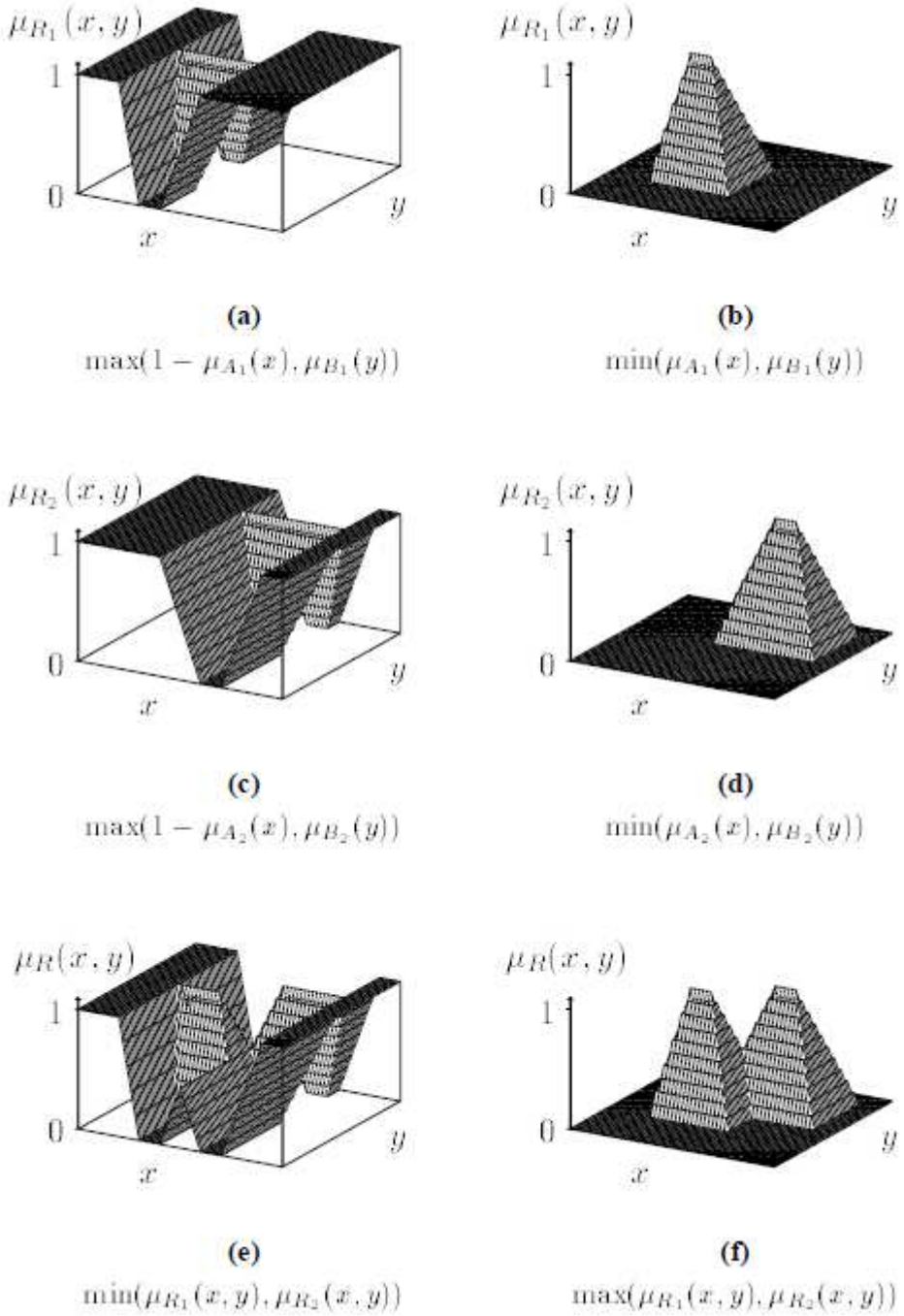


Fig.17. Agregarea regulilor fuzzy. Coloana din stânga prezintă agregarea pentru implicația Kleene-Dienes; coloana dreaptă prezintă agregarea în cazul implicațiilor Mamdani. Relația R este agregarea regulilor R_1 şi R_2.

8.2.4. Clasificarea implicațiilor fuzzy

I. Implicații fuzzy care se supun implicației clasice
$a \to b \equiv \overline{a} \vee b$ care sunt agregate cu conjuncție;

II. Implicații care se supun conjuncției clasice
$a \to b \equiv a \wedge b$ care sunt agregate cu disjuncție.

De la aceste tipuri de implicații pot apare compoziții (combinații de implicații sau implicații încuibărite):

1. implicații fuzzy bazate pe implicația din logica quantum;
$a \to b \equiv \overline{a} \vee (a \wedge b)$
2. implicații fuzzy bazate pe interpretarea modus tollens;
$a \to b \equiv \overline{b} \to \overline{a}$
3. implicații fuzzy bazate pe simetria dintre modus ponens şi modus tollens
$a \to b \equiv (a \to b) \wedge (\overline{b} \to \overline{a})$.

8.2.5. Proprietățile bazei de reguli fuzzy

In această secțiune vor fi considerate proprietățile de consistență, continuitate şi completitudine ale unei baze de reguli fuzzy.

8.2.5.1. Continuitatea unei baze de reguli

Continuitatea unei baze de reguli presupune că reguli cu premise adiacente au consecințe adiacente. Pentru un set ordonat de mulțimi fuzzy

$$A_1 < A_2 < ... < A_{i-1} < A_i < A_{i+1} < ... \qquad (39)$$

mulțimile fuzzy A_{i-1}, A_i, A_{i+1} sunt mulțimi fuzzy adiacente. Să presupunem că mulțimile fuzzy adiacente se suprapun (un bun exemplu fiind partițiile fuzzy). Premisele regulilor sunt considerate adiacente când conțin aceleaşi condiții (mulțimi fuzzy) cu excepția uneia, în acest caz mulțimile fuzzy din aceste condiții trebuie să fie adiacente.

8.2.5.2. Consistența unei baze de reguli

Aceasta se referă la consistența cunoştințelor reprezentate de baza de reguli. Un exemplu de inconsistență este dat de baza de reguli următoare unde coexistă două reguli fuzzy pentru conducerea unui robot:

Dacă obstacolul este în față *atunci* mergi la stânga;

Dacă obstacolul este în față *atunci* mergi la dreapta.

Se poate spune că este o bază de reguli greşit proiectată. Este adevărat acest lucru, însă în bazele de reguli complexe evitarea inconsistenței devine o problemă. La fel şi folosirea conectorilor *sau* pentru legarea premiselor poate duce la acelaşi tip de problemă. Un exemplu îl reprezintă următoarea bază de reguli:

Dacă x_1 este A *sau* x_2 este E *atunci* y este H

Dacă x_1 este C *sau* x_2 este F *atunci* y este I

Dacă x_1 este B *şi* x_2 este D *atunci* y este G

care duce la concluziile ambigue din tabelul 8.2.5.2.

Tabelul 8.2.5.2

x_1/x_2	A	B	C
D	H	G	I
E	H	H	H,I
F	H,I	I	I

Aşa cum se observă în tabelul anterior, regulile care conțin operatorul *sau* conduc la concluzii ambigue. Aici modelul este de dimensiuni mici şi este uşor sesizabil acest fenomen, însă în cazul unei baze de reguli mai mari şi mai complexe lucrurile devin dificile.

8.2.5.3 Completitudinea unei baze de reguli

O bază de reguli incompletă conține aşa numitele spații goale: în anumite situații ale spațiului de intrare (la nivel semantic) nu este definită nici o acțiune a spațiului de ieşire. Aceasta nu înseamnă că rezultatul inferenței unei baze de reguli incomplete nu există. O măsură a completitudinii unei baze de reguli fuzzy poate fi definită ca:

$$CM(x) = \sum_{k=1}^{N_r} \left\{ \prod_{i=1}^{N_x} \mu_{A_{i,k}}(x_i) \right\} \qquad (40)$$

unde x este un vector de date numeric. Completitudinea $CM(x)$ are o valoare mai mare ca zero în caz că una sau mai multe reguli fuzzy au vectorul de date x în suportul premisei lor. Astfel pentru valorile:

$CM(x) = 0$ bază incompletă (spațiu gol);

$0 < CM(x) < 1$ bază subcompletă;

$CM(x) = 1$ bază strict completă;

$CM(x) > 1$ bază redundantă.

In mod evident, unele regiuni din spațiul de intrare pot fi incomplete în timp ce alte regiuni pot fi redundante. Pentru reglare se recomandă ca baza de reguli să fie completă.

8.3. Raționamentul fuzzy

In această secțiune studiul se focalizează mai întâi pe inferența unei singure reguli, urmând apoi un studiu asupra regulilor fuzzy paralele.

8.3.1. Inferența unei reguli fuzzy

Inferența unei sigure reguli fuzzy este o aplicație a compoziției relațiilor fuzzy. Zadeh (1973) a introdus conceptul de regula de compoziție a inferenței „compositional rule of inference” (CRI). Vor fi tratate şi generalizări ale schemelor logice clasice modus ponens generalizat şi modus tollens generalizat.

8.3.1.1. Regula de compoziție a inferenței

Această regulă de compoziție a inferenței a fost introdusă de Zadeh (1973) şi presupune că o regulă fuzzy:

Dacă x este A *atunci* y este B

este reprezentată de o relație fuzzy R. Un rezultat B' poate fi dedus (din R aplicat lui A') prin compunerea lui A' şi R:

$$B' = A' \circ R \tag{41}$$

Relația fuzzy pentru reprezentarea regulii fuzzy poate fi oricare din implicațiile descrise în para graful 8.2.2.

8.3.1.2 Generalizarea modus ponens şi tollens

Modus ponens generalizat a fost introdus de Zadeh (1973). Este o versiune generalizată a raționamentului din logica clasică şi se bazează pe o relație de tip dacă-atunci:

Dacă x este A *atunci* y este B

x este A'

---

y este B'

A' reprezintă datele şi B' rezultatul inferenței sau $B' = A' \circ R$. Tabelul de adevăr pentru modus ponens este următorul:

A'	$A \to B$	B'
1	1	1
0	1	?

Pentru a-l rezolva utilizând CRI este necesară o relație care să reprezinte regula dacă-atunci (vezi 8.2.2). Un exemplu pentru modus ponens generalizat este următorul: presupunem regula „*dacă* este fum *atunci* este foc” şi data „este fum” atunci „este foc” poate fi obținut cu modus ponens (41).

Ca şi modus ponens, modus tollens poate fi generalizat. Schema de inferență este:

Dacă x este A *atunci* y este B

y este B'

---

x este A'

sau $A' = R \bullet B'$.

Tabelul de adevăr pentru modus tollens clasic este următorul:

B'	$A \to B$	A'
1	1	?
0	1	0

Modus tollens poate fi utilizat pentru raționamentul „*dacă* este fum *atunci* este foc” şi „nu este foc” atunci „nu este fum”.

8.3.1.3. Inferența unei reguli modelate prin implicația T

Când funcția de implicație fuzzy este o normă T, inferența poate fi simplificată în multe cazuri. De exemplu presupunând următoarea regulă

r_k : *dacă* x_1 este $A_{1,k}$ *şi* x_2 este $A_{2,k}$ *atunci* y este B_k

unde conjuncția şi implicația sunt reprezentate de o normă T.

$$B_k^{'} = T\left(A_1^{'}, A_2^1\right) \circ_T R_k = T\left(A_1^{'}, A_2^1\right) \circ_T T\left(T\left(A_{1,k}, A_{2,k}\right), B_k\right) \tag{42}$$
$$= T\left(T\left(hgt\left(T\left(A_1^{'}, A_{1,k}\right)\right), hgt\left(T\left(A_2^{'}, A_{2,k}\right)\right), B_k\right)\right)$$

unde $\circ_T$ reprezintă compoziția sup-T. In majoritatea cazurilor norma T aleasă este fie operatorul *min* fie operatorul *produs*. Simplificarea este posibilă din moment ce conjuncția, implicația şi compoziția se bazează pe aceeaşi normă T. Când operatorul *min* este utilizat pentru conjuncție şi implicație apare o simplificare majoră în inferența unei reguli fuzzy. Astfel pentru CRI, când se utilizează operatorul *min* şi conectorul *şi*, duce la:

$$\mu_{B_k^{'}}(y) = \sup_{x_1, x_2}\left\{\left(\mu_{A_1^{'}}(x_1) \wedge \mu_{A_2^{'}}(x_2)\right) \wedge \mu_{R_k}(x_1, x_2, y)\right\} \tag{43a}$$

$$= \sup_{x_1, x_2}\left\{\left(\mu_{A_1^{'}}(x_1) \wedge \mu_{A_2^{'}}(x_2)\right) \wedge \left[\left(\mu_{A_{1,k}}(x_1) \wedge \mu_{A_{2,k}}(x_2) \wedge \mu_{B_k}(y)\right)\right]\right\} \tag{43b}$$

$$= \left\{\sup_{x_1, x_2}\left[\left(\mu_{A_1^{'}}(x_1) \wedge \mu_{A_2^{'}}(x_2)\right) \wedge \left(\mu_{A_{1,k}}(x_1) \wedge \mu_{A_{2,k}}(x_2)\right)\right]\right\} \wedge \mu_{B_k}(y) \tag{43c}$$

$$= \left\{\sup_{x_1}\left(\mu_{A_1^{'}}(x_1) \wedge \mu_{A_{1,k}}(x_1)\right) \wedge \sup_{x_2}\left(\mu_{A_1^{'}}(x_2) \wedge \mu_{A_{2,k}}(x_2)\right)\right\} \wedge \mu_{B_k}(y) \tag{43d}$$

$$= hgt\left(A_1^{'} \cap A_{1,k}\right) \wedge hgt\left(A_2^{'} \cap A_{2,k}\right) \wedge \mu_{B_k}(y). \tag{43e}$$

Astfel, inferența se reduce la tăierea mulțimii fuzzy B_K din consecința regulii r_k la valoarea numerică β_k

$$\beta_k = \bigwedge_{i=1}^{N_x} \sup_{x_i} \min\left(\mu_{A_i^{'}}(x_i), \mu_{A_{i,k}}(x_i)\right) = \bigwedge_{i=1}^{N_x} hgt\left(A_i^{'} \cap A_{i,k}\right) \tag{44}$$

Valorile β_k se mai numesc valori ale suportului, grad de îndeplinire a regulii sau gradul de potrivire între date şi premisele regulii. Inferența este redusă astfel la o simplă schemă de calcul. În capitolul 9 este arătat faptul că metoda de inferență „max-min" este deseori utilizată în reglarea fuzzy. Inferența unei reguli este schematic reprezentată în fig. 18 (a,b,c).

Atunci când pentru funcția de implicație şi de conjuncție se foloseşte operatorul produs, se utilizează compoziția *sup-produs* (fig. 18 d,e,f). Mulțimea fuzzy B_K este scalată cu o valoare β_k dată de:

$$\beta_k = \bigwedge_{i=1}^{N_x} \sup_{x_i} \mu_{A_i^{'}}(x_i)\mu_{A_{i,k}}(x_i) = \bigwedge_{i=1}^{N_x} hgt\left(A_i^{'} * A_{i,k}\right) \tag{45}$$

unde * reprezintă conjuncția mulțimilor fuzzy cu operatorul *produs*. Astfel, metoda de inferență are o schemă de calcul similară celei pentru operatorul *min*

$$B_k^{'} = A^{'} \circ_* R_k = A^{'} \circ_* \left(A_k * B_k\right) = hgt\left(A^{'} * A_k\right) * B_k \tag{46}$$

unde $\circ_*$ reprezintă compoziția sup-produs. In cap.9 această schemă de inferență tipică va fi referită ca metoda de inferență „max-produs".

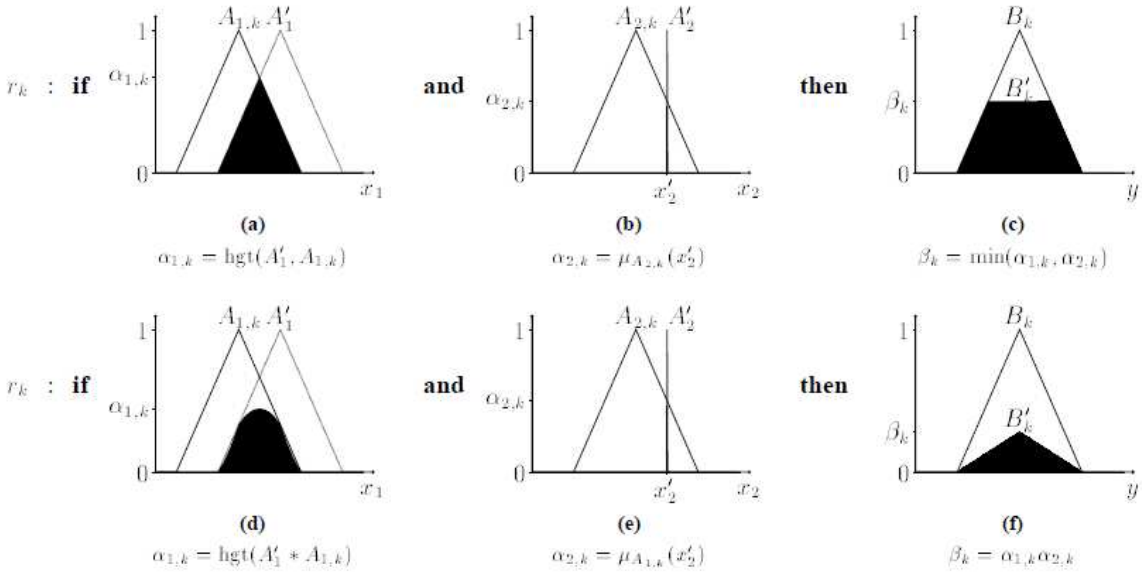


Fig.18. Inferența unei reguli cu operatorul sup-min (a,b,c) şi sup-produs (d,e,f), regula r_k.fiind: dacă x_1 este $A_{1,k}$ şi x_2 este $A_{2,k}$ atunci y este B_k.

8.3.2 Inferența unei baze de reguli

In subcapitolul 8.3.1.1 a fost explicat modul cum o regulă fuzzy reprezentată de o relație fuzzy poate fi utilizată pentru a obține date noi prin aplicarea compoziției relației ce descrie datele şi a relației ce descrie regula. In mod normal cunoştințele sunt reprezentate de un set de reguli fuzzy paralele: baza de reguli fuzzy. In acest capitol vor fi prezentate mai multe abordări pentru inferența unui set de reguli paralele.

8.3.2.1. Inferența locală versus inferența globală

Când se consideră un set de reguli (paralele), este posibilă deducerea rezultatelor din reguli individuale şi combinarea acestora în rezultate globale. Această metodă este utilizată de obicei în sistemele expert convenționale. O altă abordare este combinarea tuturor regulilor la început (agregarea lor) şi deducerea rezultatului global din această combinare. Deci avem de-a face cu două abordări:

- Inferența locală- inferența este realizată pe reguli individuale şi rezultatele sunt apoi agregate
- Inferența globală- care presupune că o relație R reprezintă baza de reguli fuzzy, iar R este agregarea relațiilor fuzzy R_k reprezentând reguli individuale.

8.3.2.2. Reguli modelate de implicații bazate pe conjuncția clasică

Când implicațiile sunt reprezentate de norme T, iar agregarea se face cu disjuncție, rezultatele inferenței locale sau globale sunt identice. Relația R, reprezentând baza de reguli este agregarea relațiilor R_k :

$$R = \bigcup_k R_k \tag{47}$$

unde R_k este relația fuzzy reprezentând regula r_k. Aplicarea CRI setului de reguli poate fi simplificată astfel

$$B' = A' \circ R = A' \circ \left\{ \bigcup_k R_k \right\} = \bigcup_k \left\{ A' \circ R_k \right\} = \bigcup_k B'_k \qquad (48)$$

Astfel s-a demonstrat că rezultatul inferenței globale este acelaşi cu cel al inferenței locale.

8.3.2.3 Reguli modelate de implicațiile bazate pe implicații clasice

In acest caz, conjuncția va fi utilizată pentru agregarea relațiilor fuzzy individuale R_k pentru obținerea relației fuzzy globale R. Presupunând utilizarea operatorului *min* pentru agregare, deducerea unei mulțimi fuzzy B' dintr-o relație R şi datele A' duce la:

$$B' = A' \circ R = A' \circ \left\{ \bigcap_k R_k \right\} \subseteq \bigcap_k \left\{ A' \circ R_k \right\} \qquad (49)$$

De aici rezultă clar că rezultatul inferenței globale 49 poate fi diferit de cel al inferenței locale c. Următorul exemplu arată ineficiența inferenței locale (Dubois şi Prade, 1991). Presupunând două reguli $A_1 \to B_1$ şi $A_2 \to B_2$ şi datele $A' = A_1 \cup A_2$ unde A_1, A_2, B_1 şi B_2 sunt submulțimi clasice pe universul de intrare X şi respectiv de ieşire Y.

Atunci inferența globală duce la

$$A' \circ R = B_1 \cup B_2 \qquad (50)$$

în timp ce inferența locală va duce la

$$\left\{ A' \circ R_1 \right\} \cap \left\{ A' \circ R_2 \right\} = Y$$

iar rezultatul Y este necunoscut şi într-adevăr (49) se verifică în sensul că $B_1 \cup B_2 \subseteq Y$. Oricum utilizând inferența locală, rezultatul obținut nu conține informație B' fiind necunoscut. Utilizând inferența globală, rezultatul este disjuncția lui B_1 şi B_2, care este şi rezultatul corect.

In general, rezultatele deduse în urma inferenței locale sunt mai puțin restrictive ca cele obținute prin inferența globală. Rezultatele inferenței locale nu sunt greşite, ci conțin mai puțină informație decât ar putea să conțină cele asociate inferenței globale bazată pe datele şi cunoştințele disponibile.

9. Reglarea fuzzy

In acest capitol este prezentat modul de lucru al regulatoarelor fuzzy. Este prezentată o abordare teoretică a acestui domeniu în 9.1., urmată de abordarea practică în 9.2. In subcapitolul 9.3 se prezintă cele două tipuri de reguli fuzzy care se utilizează în mod curent.

9.1. Abordarea teoretică

Teoretic, reglarea fuzzy reprezintă aplicarea regulii compoziționale de inferență CRI. Dându-se o relație R, reprezentând regulatorul şi o relație A' reprezentând intrarea în regulator, ieşirea fuzzy B' poate fi obținută prin compunerea:

$$B' = A' \circ R \qquad (51)$$

Totuşi, pentru că intrările şi ieşirile regulatorului fuzzy sunt în mod normal valori numerice, este necesară o transformare din intrare numerică în intrare fuzzy respectiv o transformare de la o ieşire fuzzy la o ieşire numerică. Prima transformare se numeşte fuzificare, cea din urmă defuzificare. O reprezentare schematică este dată în fig. 19.

Algoritmul de reglare este reprezentat de regulile fuzzy. De exemplu, regulile clasifică intrările regulatorului în funcție de premisele regulilor şi au drept consecință o incrementare/decrementare a ieşirii regulatorului.

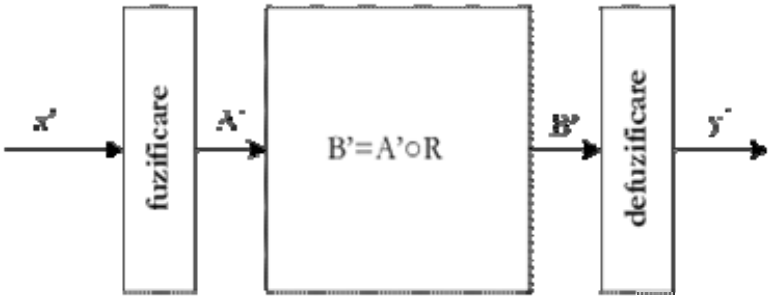


Fig. 19. Reprezentarea schematică a regulatorului fuzzy. Valoarea numerică x' este fuzificată în A'. Ieşirea fuzzy B' este defuzificată în ieşirea numerică y'.

9.1.1. Fuzificarea intrărilor

In abordarea teoretică, faza de fuzificare a unui regulator fuzzy este de fapt construirea unei relații fuzzy de intrare.

Relația fuzzy de intrare A' este conjuncția a N_x mulțimi de intrare fuzzy A_i' unde N_x reprezintă numărul de intrări ale regulatorului. Mulțimile fuzzy A_i' sunt reprezentări fuzzy ale intrărilor regulatorului fuzzy x_i':

$$A_i' = fuzz(x_i') \qquad (52)$$

unde *fuzz* este funcția de fuzificare: un operator care transformă o valoare numerică într-o reprezentare de tip mulțime fuzzy. Dacă intrarea x_i este o valoare numerică, mulțimea fuzzy A_i' este dată de exemplu de un singleton:

$$\mu_{A_i^{'}}(x_i) = \begin{cases} 1 & daca\ x_i = x_i^{'} \\ 0, & in\ rest \end{cases} \tag{53}$$

Dacă intrările prezintă imprecizii sau zgomote, elemente necunoscute, acestea pot fi fuzificate utilizând numere fuzzy. Relația fuzzy de intrare este determinată prin combinarea mulțimilor fuzzy pentru fiecare intrare:

$$A^{'} \int_{X_1 \times ... \times X_{N_x}} \left(\underset{i=1}{\overset{N_x}{T}} \mu_{A_i^{'}}(x_i) \right) / (x_1, ..., x_{N_x}) \tag{54}$$

unde T este norma T care realizează conjuncția în premisă. Datele de intrare sunt în fapt reprezentate de $A_1^{'}$ *şi* $A_2^{'}$ *şi...şi* $A_i^{'}$ *şi* $...A_{N_x}^{'}$.

9.1.2. Defuzificarea ieşirii

Defuzificarea este necesară pentru transformarea ieşirii fuzzy a unui regulator fuzzy într-o reprezentare numerică. Din punct de vedere teoretic, ieşirea fuzzy este reprezentată de o relație fuzzy. Pentru defuzificarea relațiilor fuzzy sunt utilizate în mod curent 3 metode: metoda centrului de greutate, mean of maxima, si a centrului suprafeței.

9.1.2.1. Defuzificarea cu metoda centrului de greutate (COG)

Metoda este aceeaşi cu cea utilizată pentru calculul centrului de greutate al unui sistem de puncte materiale. Diferența este că punctele materiale sunt înlocuite cu valori de apartenență. Pentru mulțimile fuzzy unidimensionale metoda se mai numeşte a centrului suprafeței.

Metoda se defineşte prin operatorul *cog* (center of gravity) astfel:

$$cog(B^{'}) = \frac{\int_Y \mu_{B^{'}}(y) y dy}{\int_Y \mu_{B^{'}}(y) dy} \tag{55}$$

iar varianta discretă prin

$$cog(B^{'}) = \frac{\sum_{q=1}^{N_q} \mu_{B^{'}}(y_q) y_q}{\sum_{q=1}^{N_q} \mu_{B^{'}}(y_q)} \tag{56}$$

unde N_q este numărul cuantelor utilizate pentru discretizarea funcției de apartenență $\mu_{B^{'}}(y)$ a ieşirii fuzzy $B^{'}$. In fig. 20 sunt date exemple de aplicare a acestei metode.

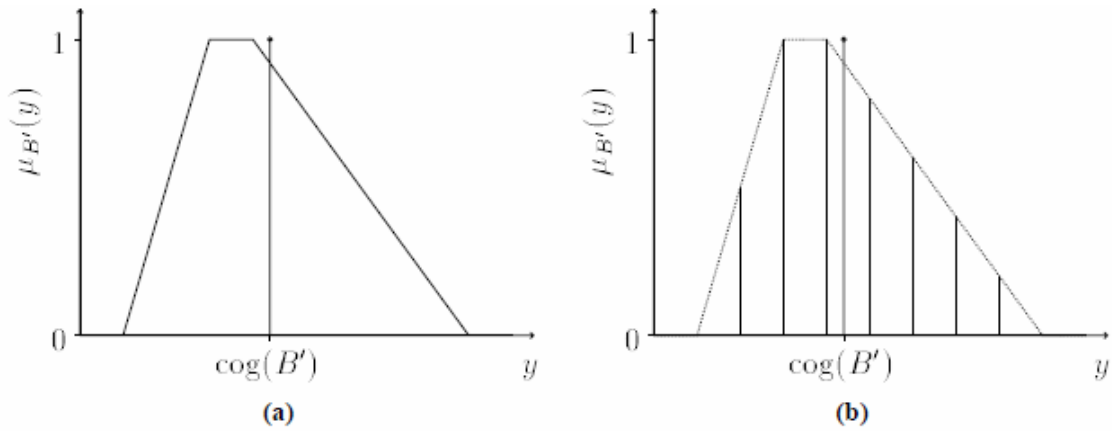


Fig. 20. Exemple de aplicare a defuzificării cu metoda COG pentru cazul continuu (a) şi discret (b).

Metoda nu este limitată la mulțimile fuzzy unidimensionale, se poate aplica şi relațiilor fuzzy. Totuşi în practică, această metodă este utilizată preponderent în cazul mulțimilor fuzzy unidimensionale.

9.1.2.2. Metode indexate de defuzificare (idfz)

Acestea sunt utilizate pentru a determina acea parte din ieşirea fuzzy ale cărei valori de apartenență sunt sub un anumit prag β_t:

$$idfz(B', \beta_t) = dfz(B' \cap \alpha - cut(B', \beta_t)) \text{ cu } \alpha = \beta_t$$
$$= dfz(B' \cap \beta_t - cut(B')) \qquad (57)$$

unde *dfz* este metoda de defuzificare, iar *idfz* este varianta ei indexată. Metoda de defuzificare dfz este aplicată numai unei părți din ieşirea fuzzy, care are gradul de apartenență mai mare sau egal cu β_t. In fig. 21(a) este dat un exemplu de metodă indexată a centrului de greutate cu $\beta_t = \frac{1}{2}$.

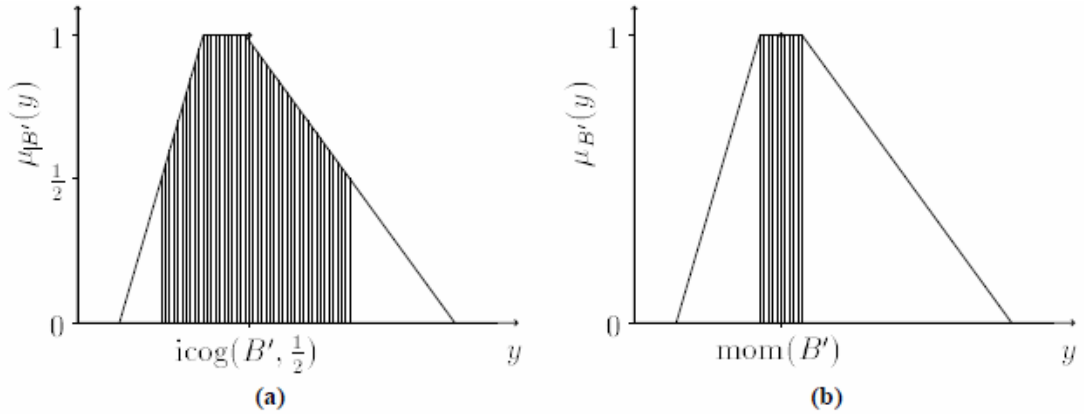


Fig.21. Exemplu de metodă COG indexată (a) şi MOM (b).

9.1.2.3. Defuzificarea mean of maxima (MOM)

In afară de metoda centrului de greutate, o altă metodă de bază de defuzificare este mean of maxima (MOM) definită prin:

$$mom(B') = icog(B', hgt(B')) = cog(B' \cap \alpha - cut(B', hgt(B'))) \qquad (58)$$

unde *icog* este versiunea indexată a *cog* definită cu (55). In fig. 21 b este dat un exemplu de determinare $mom(B')$. Această metodă de defuzificare ignoră o mare parte din informația furnizată de mulțimea fuzzy, datorită aplicării tăieturii $\alpha - cut$ cu $\alpha = hgt(B')$.

9.1.2.4. Metoda de defuzificare a centrului suprafeței (COA – center of area)

Această metodă în literatură este de cele mai multe ori confundată cu COG. Metoda este definită prin:

$$\int_{\inf_Y y}^{coa(B')} \mu_{B'}(y)dy = \int_{coa(B')}^{\sup_Y y} \mu_{B'}(y)dy \qquad (59)$$

valoarea numerică $y_{coa}(B')$ împarte aria funcției de apartenență în două părți egale (fig.22).

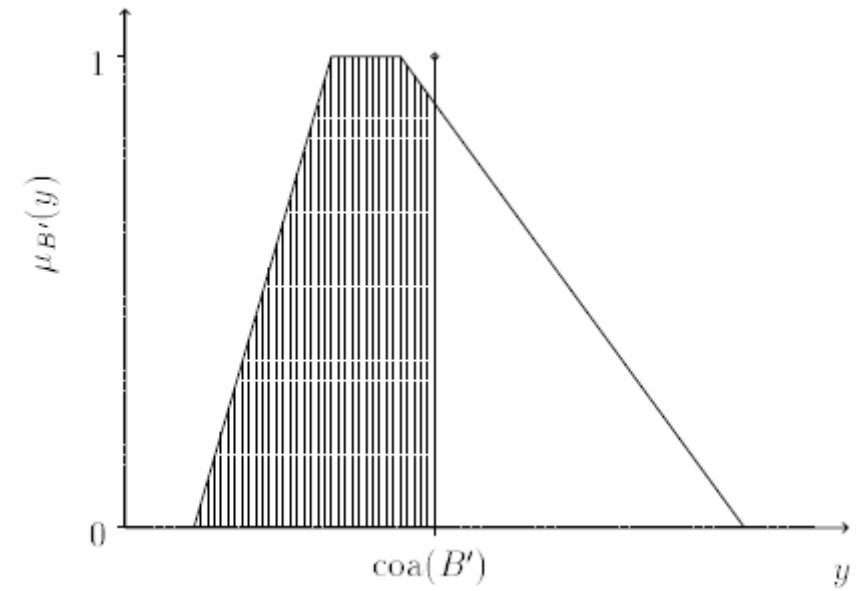


Fig. 22. Exemplu de defuzificare cu metoda centrului suprafeței.

De exemplu pentru mulțimea fuzzy din fig.23, se obțin prin defuzificarea cu metodele prezentate următoarele rezultate, prezentate în tabelul 9.1.2:

Tabelul 9.1.2.

Metoda	Dfz(A)	Idfz(A, 1/2)
COG continuă	$4\frac{4}{9}$	$4\frac{4}{29}$
COG discretă	$4\frac{4}{9}$	4
COA	$4\frac{1}{2}$	$4\frac{1}{16}$
MOM	$3\frac{1}{2}$	$3\frac{1}{2}$

9.1.3. Exemplu de abordare teoretică

Regulatorul fuzzy se bazează pe abordarea teoretică descrisă în secțiunile anterioare. El poate fi considerat un sistem de tip SISO şi totodată un regulator P neliniar

atunci când intrarea regulatorului este eroarea (diferența între valoarea dorită şi cea actuală a ieşirii procesului).

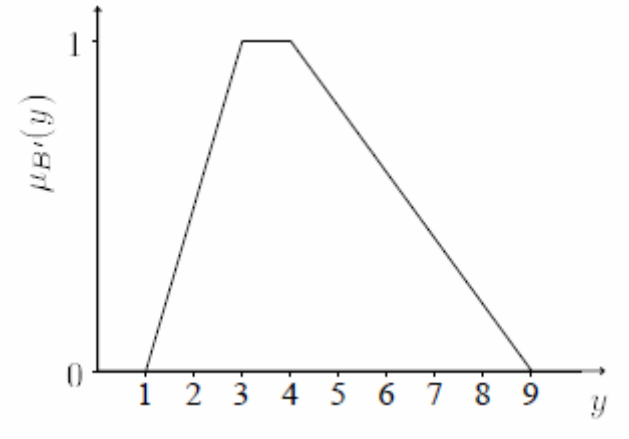


Fig.23. Funcția de apartenență a mulțimii fuzzy B'.

Mai întâi trebuie alese mulțimile fuzzy pentru intrarea şi ieşirea regulatorului. In fig.24 sunt prezentate mulțimile fuzzy pentru intrare (24a) şi pentru ieşire (24b).

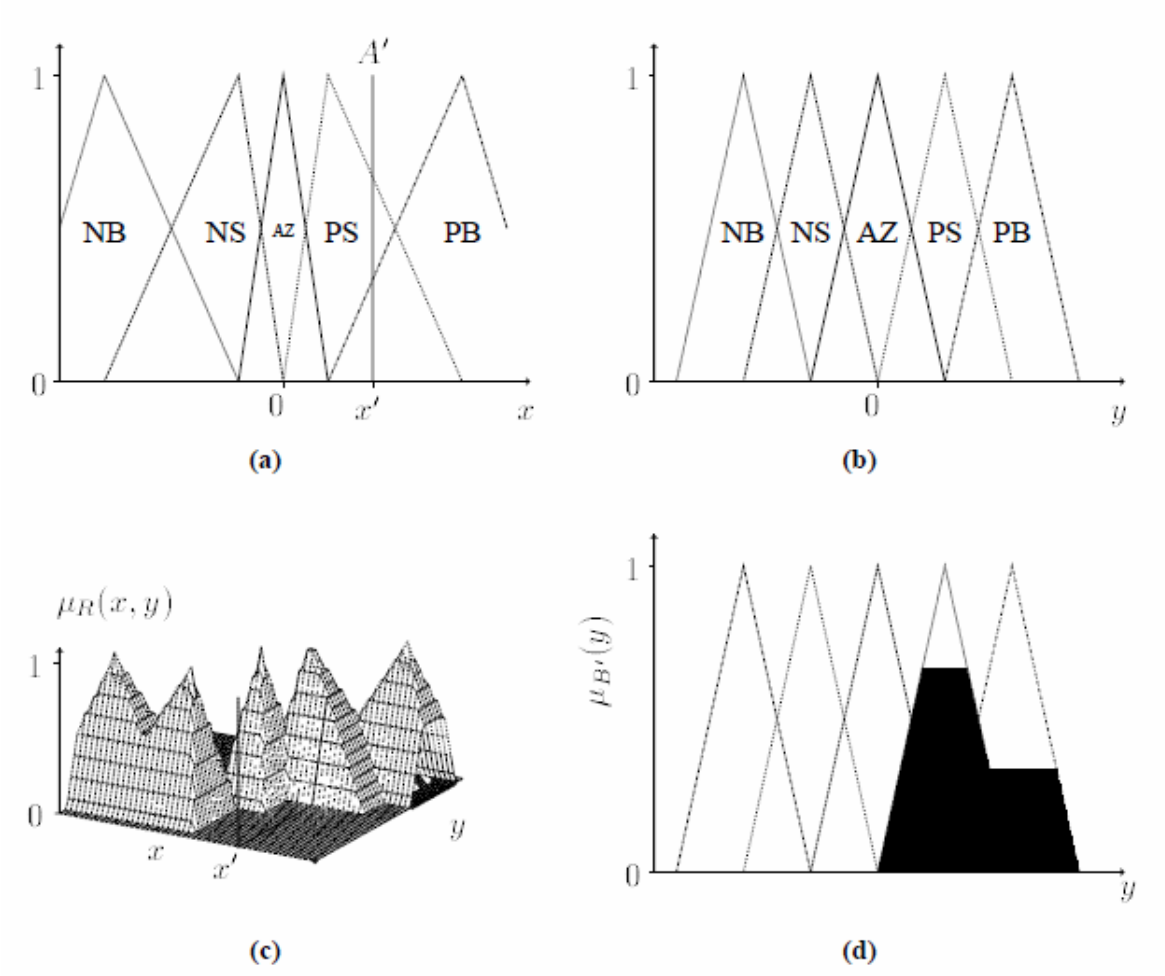


Fig.24. Exemplu de abordare teoretică în reglarea fuzzy. (a) şi (b) reprezintă mulțimile fuzzy pentru universul de intrare şi respectiv de ieşire. Relația R a regulatorului fuzzy rezultată (c) este aplicată pentru relația fuzzy de intrare A' (un singleton) iar rezultatul acestei compoziții este prezentat în (d).

Alegerea e oarecum arbitrară, un număr de la 5 la 9 mulțimi fuzzy pe univers fiind însă utilizat în mod curent în reglarea fuzzy. Pasul următor în proiectarea regulatorului fuzzy constă în stabilirea bazei de reguli. Baza de reguli conține următoarele reguli:

r_1 : dacă x este NB atunci y este NB

r_2 : dacă x este NS atunci y este NB

r_3 : dacă x este AZ atunci y este AZ

r_4 : dacă x este PS atunci y este PS

r_5 : dacă x este PB atunci y este PB

unde AZ- about zero, aproape zero
NB- negative big, negativă mare
NS- negative small, negativă mică
PB- positive big, pozitivă mare
PS- positive small, pozitivă mică

Pentru fiecare regulă trebuie construită o relație fuzzy R_k. Relațiile R_k sunt agregate pentru obținerea relației R a regulatorului fuzzy. Relația fuzzy R este reprezentată în fig.24c. Urmează defuzificarea. Metoda COG este utilizată în acest exemplu. Proiectarea regulatorului fuzzy este completă în acest moment, în condițiile în care cunoştințele reprezentate sub formă de reguli se presupun a fi corecte. Pentru determinarea ieşirii regulatorului fuzzy la o intrare numerică x' ca în fig. 24a se foloseşte CRI, adică compunerea relației A' (un singleton în acest caz) cu relația regulatorului R. Acest rezultat fuzzy trebuie defuzificat pentru a se obține o valoare numerică la ieşirea regulatorului (fig. 24d).

Obs. Implementarea regulatorului fuzzy bazat pe abordarea teoretică necesită discretizarea relațiilor fuzzy pentru stocarea lor în memoria calculatorului. Pentru minimizarea erorilor datorate discretizării, pasul de discretizare trebuie ales suficient de mic.

In exemplul ales, rezultatul B' poate fi obținut şi prin:

$$\mu_{B'}(y) = \max\left(\min\left(\mu_{PS_x}(x'), \mu_{PS_y}(y)\right), \min\left(\mu_{PB_x}(x'), \mu_{PB_y}(y)\right)\right) \tag{60}$$

Este practic cea mai utilizată metodă în reglarea fuzzy.

9.2. Abordarea practică în reglarea fuzzy

In abordarea teoretică s-a arătat că algoritmul fuzzy de reglare constă în 3 faze: fuzificare, compunerea între relația fuzzy de intrare şi relația fuzzy a regulatorului şi defuzificarea. Aceasta implică un efort mare de calcul în cazul funcțiilor multidimensionale, ceea ce nu este recomandat în practică. In practică, reglarea fuzzy implică utilizarea inferențelor locale. In cele ce urmează intrările vor fi considerate numerice (singletonuri) din motive de simplitate. Această simplificare nu reprezintă o restricție severă din moment ce în mod obişnuit intrările în regulator sunt în mod obişnuit valori numerice preluate de la senzori.

9.2.1. Inferența fuzzy în practică

In practică, inferența unei baze de reguli este locală. Astfel, inferența unei baze de reguli presupune inferența fiecărei reguli urmată de agregarea rezultatelor obținute. Pentru intrări numerice rezultatul inferenței globale este identic cu cel al inferenței locale. In cele ce urmează va fi prezentată o schemă practică de inferență fuzzy precum şi o analiză a diferitelor implicații fuzzy utilizate în reglarea fuzzy.

9.2.1.1. Schema de inferență fuzzy în practică

Inferența în reglarea fuzzy este reprezentată de paşii următori:

1. potrivirea propozițiilor fuzzy de tipul x este $A_{i,k}$, utilizate în premisele regulilor r_k cu datele numerice de intrare $x_i^{'}$ ale regulatorului fuzzy;

$$\alpha_{i,k} = \mu_{A_{i,k}}\left(x_i^{'}\right)$$

unde $\alpha_{i,k}$ este valoarea numerică ce reprezintă potrivirea. In cazul intrărilor fuzzy $A_i^{'}$ potrivirea este în mod normal reprezentată de

$$\alpha_{i,k} = hgt(A_i^{'} \cap A_{i,k}) .$$

2. determinarea gradelor de îndeplinire pentru fiecare regulă r_k :

$$\beta_k = \mathop{T}_{i=1}^{N_x} \alpha_{i,k}$$

unde T este norma T reprezentând conectorul ***şi*** în premisele regulilor. Dacă se foloseşte conectorul ***sau***, norma T este înlocuită cu norma S.

3. determinarea rezultatului $B_k^{'}$ pentru fiecare regulă r_k :

$$\mu_{B_k^{'}}(y) = I\left(\beta_k, \mu_{B_k}(y)\right)$$

unde I este implicația utilizată pentru modelarea regulilor fuzzy. Acestea pot fi implicații bazate pe conjuncția clasică sau pe implicația clasică.

4. Agregarea rezultatelor $B_k^{'}$ ale regulilor individuale r_k pentru obținerea rezultatului global $B^{'}$:

$$\mu_{B^{'}}(y) = \begin{cases} \bigcup_k \mu_{B_k^{'}}(y) & \text{pentru implicatii bazate pe conjunctia clasicã} \\ \bigcap_k \mu_{B_k^{'}}(y) & \text{pentru implicatii bazate pe implicatia clasicã} \end{cases}$$

Aşa cum se poate observa din această schemă de inferență fuzzy practică nu se construieşte o relație fuzzy de intrare (în cadrul fuzificării). In 9.2.2 va fi discutat modul de

potrivire a propozițiilor fuzzy cu datele disponibile care reprezintă faza de fuzificare în aplicațiile reglării fuzzy.

9.2.1.2. Inferența cu implicații T

In 8.3.2.2. s-a demonstrat că utilizarea normelor T duce la simplificarea inferenței şi mai mult, rezultatul inferenței locale coincide cu cel al inferenței globale:

$$A' \circ \left\{ \bigcup_k R_k \right\} = \bigcup_k \left\{ A' \circ R_k \right\}$$

Considerând intrările regulatorului de tip singleton (valori numerice), inferența fuzzy pentru fiecare regulă este redusă la potrivirea datelor cu premisa regulii:

$$\mu_{B'}(y) = T\left(\beta_k, \mu_{B_k}(y)\right) \text{ cu } \beta_k = \overset{N_x}{\underset{i=1}{T}} \mu_{A_{i,k}}\left(x_i'\right)$$

unde x_i' sunt intrările regulatorului fuzzy. Astfel fiecare rezultat B_k este restricționat la valoarea β_k prin intermediul normei T care reprezintă implicația fuzzy. Ieşirea fuzzy se obține prin agregarea subrezultatelor cu operatorul max:

$$\mu_{B'}(y) = \max_k \mu_{B_k'}(y).$$

Acest rezultat permite găsirea unei căi simple şi comode de calcul a ieşirii regulatorului fuzzy analitic, fără calcule prea complicate şi încărcarea memoriei. Această metodă este utilizată în aplicațiile reglării fuzzy. Un exemplu simplu este dat de regula următoare:

Dacă eroarea este mică **şi** frecvența de modificare a erorii este mare **atunci** reduce comanda care poate fi o parte dintr-un regulator fuzzy de tip PI. Avându-se în vedere literatura de specialitate se poate afirma că cele mai utilizate sunt implicațiile T.

9.2.1.3. Inferența cu implicații S

Implicațiile S sunt cele bazate pe implicația clasică. S-a demonstrat că rezultatele inferenței locale sunt egale cu cele ale inferenței globale doar în cazul în care intrările nu sunt fuzzy. Ieşirea fuzzy a regulatorului este determinată de:

$\mu_{B'}(y) = \min_k \mu_{B_k'}(y) = \min_k S\left(1 - \beta_k, \mu_{B_k}(y)\right)$ unde $\beta_k = \overset{N_x}{\underset{i=1}{T}} \mu_{A_{i,k}}\left(x_i'\right)$ şi x_i' este intrarea numerică. Acest rezultat poate părea la fel de simplu ca în cazul implicațiilor de tip T, însă aplicarea implicațiilor S în reglarea fuzzy poate conduce la rezultate nedorite. Implicațiile S nu sunt potrivite pentru aplicarea lor în reglarea fuzzy din cauza faptului că ele pot duce la ieşiri nedeterminate fuzzy. O altă proprietate importantă este că inferența decide în favoarea regulii cu gradul cel mai mare de îndeplinire, ceea ce poate duce la tranziții discontinue de la o ieşire numerică a regulatorului la următoarea.

9.2.2. Fuzificarea intrării

In abordarea practică a reglării fuzzy, fuzificarea nu reprezintă construirea relației fuzzy de intrare. Faza de fuzificare constă în determinarea gradului de potrivire dintre intrările regulatorului şi mulțimile fuzzy care reprezintă etichetele lingvistice de pentru intrări în premisele regulilor. Nu este o fază de preprocesare ca în cazul abordării teoretice, la evaluarea premisei unei reguli, valorile gradelor de potrivire necesare putând fi calculate rapid. Aceasta previne calcularea nedorită a valorilor de potrivire din moment ce acestea vor fi evaluate doar atunci când este nevoie. In paragraful anterior au fost menționate

câteva din dezavantajele utilizării implicațiilor S în reglarea fuzzy. Avându-se în vedere aceste dezavantaje, fuzificarea va fi considerată în continuare doar pentru cazurile în care regulile fuzzy sunt modelate cu implicații T.

Din moment ce fuzificarea în abordarea practică reprezintă determinarea gradului de potrivire între intrări şi etichetele lingvistice utilizate în premisele regulilor, „fuzificarea" în cazul implicațiilor T este în mod normal determinată de:

$$\alpha_{i,j} = hgt\left(A_i^{'} \cap A_{i,j}\right)$$

unde $\alpha_{i,j}$ reprezintă gradul de potrivire dintre datele $A_i^{'}$ pentru intrarea x_i şi mulțimea j fuzzy $A_{i,j}$ a universului de discuție x_i. In secțiunea 9.2.3 vor fi descrise câteva tipuri de metode practice de inferență utilizate în mod frecvent. Pentru anumite tipuri de inferență este arătat că anumite combinații ale potrivirii (paşii 1 şi 2) şi ale modificării (pasul 3) nu corespund compunerii relațiilor fuzzy din abordarea teoretică.

In fig. 25 sunt prezentate câteva „potriviri" între intrările fuzzy şi mulțimile fuzzy definite pentru o intrare specifică. Potrivirea intrărilor numerice cu premisele regulilor este un caz particular de potrivire a unei intrări fuzzy cu premisele regulilor.

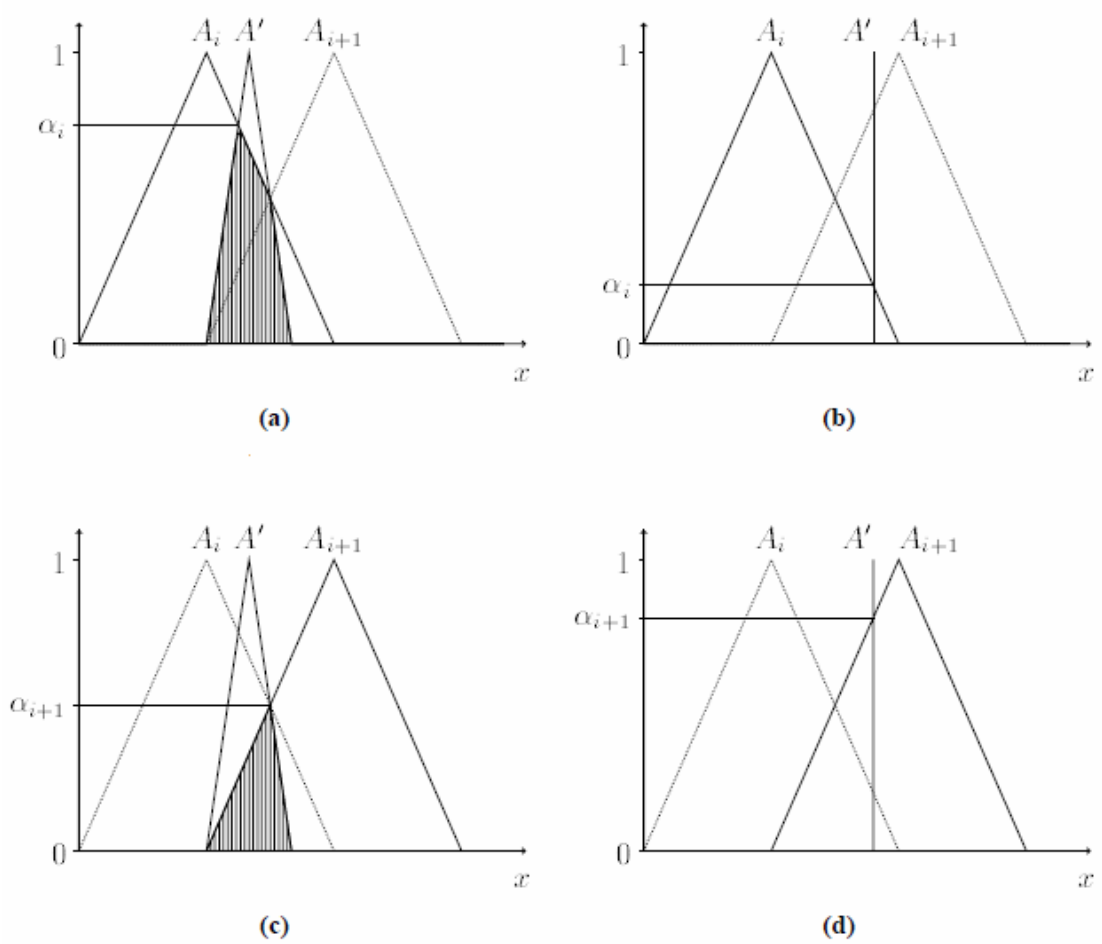


Fig.25. Fuzificarea practică atunci când se utilizează implicații T pentru regulile fuzzy:potrivirea dintre intrările (fuzzy) şi mulțimile fuzzy reprezentate de etichetele lingvistice utilizate în premisele regulilor. Potrivirea pentru intrări fuzzy (a) şi (c),şi pentru intrări numerice (b) şi (d).

Uneori fuzificarea în reglarea fuzzy poate fi văzută ca transformarea intrării într-un vector care să conțină gradele de potrivire cu fiecare etichetă lingvistică din universul de

discuție al intrării. Pentru exemplul considerat în fig.20 poate fi construit următorul vector α care să conțină gradele de potrivire

$$\alpha = [0...0\ \alpha_i\ \alpha_{i+1}\ 0\ ...0]$$

unde α_i reprezintă potrivirea dintre datele A' şi mulțimea fuzzy A_i din universul de discuție a lui x:

$$\alpha_i = \begin{cases} hgt(A' \cap A_i) \text{ in general} \\ \mu(A_i(x')) \text{ pentru intrari numerice (singleton)} \end{cases}$$

Totuşi, daca se utilizează modificatori de mutare în premisele regulilor fuzzy metoda aceasta nu poate fi aplicată pentru ca în acest caz nu există un număr fix de mulțimi fuzzy A_i pe universul de discuție al intrării care să poată fi utilizate în premisele regulilor fuzzy (pentru ca modificatorii de mutare nu operează asupra funcțiilor de apartenență, ci asupra domeniului). Dacă în schimb se utilizează modificatorii ponderați, metoda de fuzificare este într-adevăr aplicabilă, pentru că intrările regulatorului sunt numerice, iar operatorii ponderați operează asupra funcțiilor de apartenență.

Astfel, valoarea α_{i,m_p} reprezentând potrivirea dintre intrarea numerica x' şi propoziția „x este $m_{mp}(A_i)$“ este determinată de:

$$\alpha_{i,m_p} = \mu_{mp(A_i)}(x') = \mu^p_{A_i}(x') = \alpha^p_i$$

unde m_p este modificatorul lingvistic ponderat.

9.2.3. Metode de inferență “obişnuite”

În secțiunile următoare vor fi considerate cele mai utilizate metode de inferență in reglarea fuzzy.

9.2.3.1. Metoda max-min

Regulatorul fuzzy introdus de Assilian şi Mamdami (1974) utilizează aşa-numita metodă max-min. Alegând operatorul *min* pentru conjuncție în premisele regulii cât şi pentru funcția de implicație şi operatorul *max* pentru agregare, aplicarea CRI duce la:

$$\mu_{B'}(y) = \overset{agregare}{\max_{k}} \; \underset{implicatie}{\min} \left(\beta_k, \mu_{B_k}(y)\right) \qquad \text{unde} \qquad \beta_k = \overset{conjunctie\ in\ premisa}{\min_{i}} \alpha_{i,k},$$

$$\alpha_{i,k} = \underbrace{\overset{proiectie}{\sup_{x}} \; \underset{combinare}{\min}}_{compozitie} \left(\mu_{A_i'}(x_i), \mu_{A_{i,k}}(x_i)\right).$$

Conjuncția, implicația şi compoziția trebuie să se bazeze pe aceeaşi normă T pentru a se obține o soluție analitică simplă în urma inferenței fuzzy. In figura 26 este prezentată inferența fuzzy bazată pe această metodă max-min. Una din intrările regulatorului este o mulțime fuzzy (A_1') şi cealaltă este intrare numerică (A_2'). Ieşirea fuzzy reprezintă agregarea (*max*) a două mulțimi fuzzy tăiate.

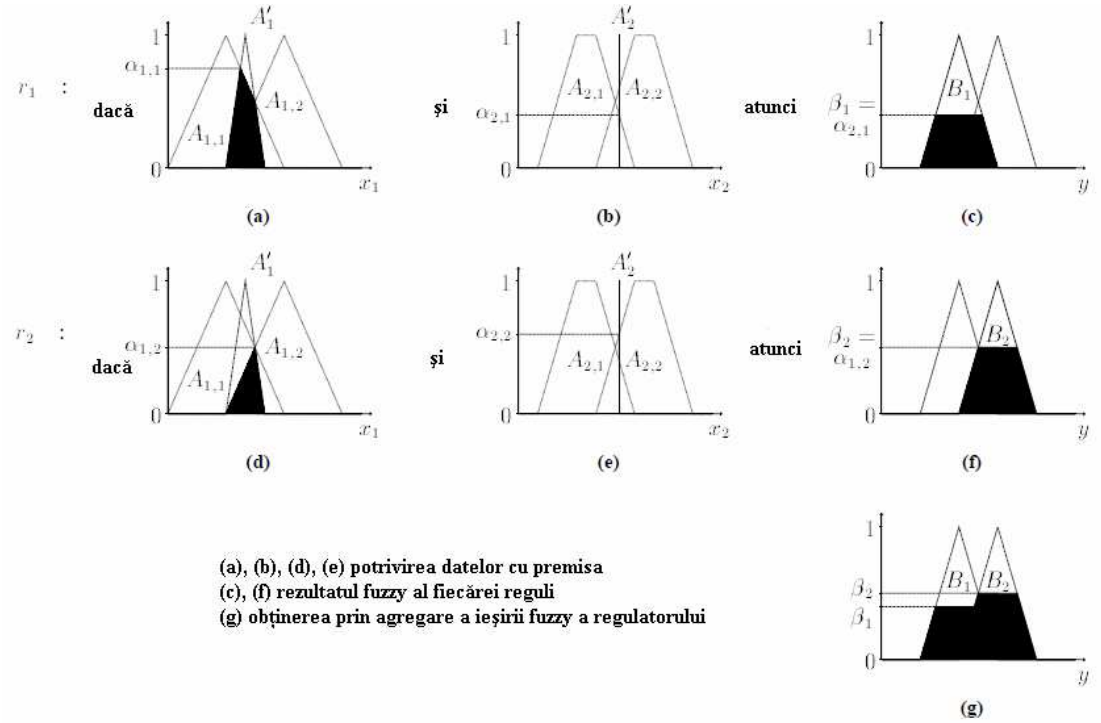


Fig.26. Metoda max-min în practică: operatorul *min* e utilizat pentru conectorul *şi* şi implicație: compoziția sup-min e utilizată pentru CRI, există două reguli: **dacă** x_1 este $A_{1,k}$ **şi** x_2 este $A_{2,k}$ **atunci** y este B_k, k=1,2.

9.2.3.2.Metoda max-prod

Este o altă metodă frecvent utilizată în reglarea fuzzy. Metoda este caracterizată prin scalarea consecinței B_k a regulii fuzzy r_k cu gradul de îndeplinire β_k al regulii şi

agregarea acestor rezultate $B_k^{'}$ pentru obținerea ieşirii regulatorului fuzzy cu ajutorul operatorului **max**: $\mu_{B'}(y) = \overset{agregrare}{\max_k} \beta_k \underset{implicatie}{*} \mu_{B_k}(y)$, unde * reprezintă înmulțirea. Agregarea este identică cu cea din metoda precedentă.

Există două variante ale acestei metode cu privire la determinarea valorii suportului β_k. Poate fi utilizat fie operatorul **min** fie **produs** pentru combinarea valorilor de potrivire $\alpha_{i,k}$ şi astfel pentru a reprezenta conjuncția în premisele regulii:

$$\beta_k = \begin{cases} \min_i \alpha_{i,k} \\ \prod_i \alpha_{i,k} \end{cases}$$

Dacă intrările regulatorului sunt numerice nu există probleme din moment ce $hgt(A_i' \cap A_{i,k}) = hgt(A_i' * A_{i,k})$

unde $A_i^{'}$ este singletonul care reprezintă intrarea numerică $x_i^{'}$. In fig.27. este dat un exemplu de ieşire fuzzy utilizând metoda de inferență max-produs.

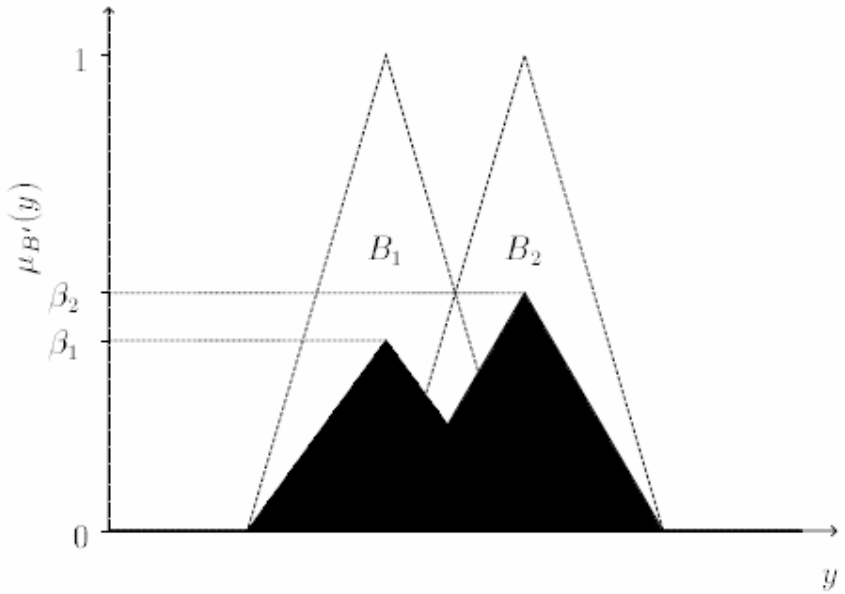


Fig.27. Metoda max-prod.

9.2.3.3.Metoda sumă-produs

Utilizează operatorul **produs** în acelaşi mod ca metoda max-produs. Ceea ce este diferit la această metodă este modul de agregare a rezultatelor. Agregarea se face cu operatorul **sumă**, dar şi aici apar variante care vor fi descrise în continuare:

- agregarea subrezultatelor $B_k^{'}$ ale tuturor regulilor fuzzy r_k sunt realizate prin adunare $\mu_{B'}(y) = \sum_k \beta_k \mu_{B_k}(y)$

Pentru anumite mulțimi fuzzy B_j aceasta poate conduce la valori $\beta_j > 1$, rezultând o mulțime fuzzy anormală care nu corespunde teoriei mulțimilor fuzzy.

Datorită defuzificării, necesară în reglarea fuzzy, aceasta este o problema minoră din punctul de vedere al abordării practice. Poate fi utilizată o sumă limitată

$$\mu_{B'}(y) = \min(\sum_k \beta_k \mu_{B_k}(y),1)$$

care elimină posibilitatea obținerii mulțimilor fuzzy anormale şi rezultatul este în conformitate cu teoria mulțimilor fuzzy.

- numai rezultatele fuzzy ale regulilor cu consecințe egale sunt agregate prin însumare, după care totul este agregat cu operatorul **max**:

$$\mu_{B'}(y) = \max_j \left\{ \sum_{k_j} \beta_{k_j} \right\} \mu_{B_j}(y)$$

unde k_j este utilizat pentru identificarea regulilor care au drept consecință $B_k = B_j$. Şi în acest caz poate fi utilizată o sumă limitată $\min\left(\sum_{kj} \beta_{kj},1\right)$. Această metodă este utilizată de obicei în combinație cu defuzificarea **fuzzy-mean** care elimină faza de agregare dacă toate regulile sunt considerate separat, neglijând echivalența consecințelor. Defuzificarea fuzzy-mean este în fapt o sumă ponderată şi poate fi considerată un caz particular al metodei COG. Această metodă de defuzificare este descrisă în secțiunea următoare.

9.2.4.Defuzificarea în practică

In 8.1.2 au fost descrise metodele COG,MOM,COA de defuzificare . O parte din acestea vor fi reluate în discuție, însă dintr-un punct de vedere practic. In acest subcapitol vor fi considerate doar ieşiri fuzzy unidimensionale.

9.2.4.1.Metode de defuzificare de mediere

In practică, se utilizează simplificări ale metodei COG. Versiunea discretizată a acestei metode a fost prezentată în 8.1.2.1. şi această metodă este frecvent utilizată în reglarea fuzzy.

Când se aplică metoda centrului de greutate, rezultatul defuzificării este o funcție neliniară care are drept variabile intrările regulatorului (în cele mai multe cazuri). Regulatorul fuzzy este în sine un regulator neliniar. Neliniaritatea este datorată în principal agregării cu operatorul **max** în combinație cu metoda COG. Pentru exemplificarea celor afirmate, sunt date două exemple în fig.28, unde intrarea numerică x' variază de la a_1 la a_2. In exemplul dat în figura 28a $\beta_1 = 1 - \beta_2$.

Când este necesar ca ieşirea regulatorului sa fie o funcție liniară de intrarea regulatorului (pentru exemplul dat în figura 28) se aleg pentru norma T şi norma S operator **produs** şi respectiv **sumă**, din moment ce în acest caz:

$$y' = \frac{\int_y \{\beta_1 \mu_{B_1}(y) + \beta_2 \mu_{B_2}(y)\} y dy}{\int_y \{\beta_1 \mu_{B_1}(y) + \beta_2 \mu_{B_2}(y)\} dy} = \frac{\beta_1 \int_y \mu_{B_1}(y) y dy + \beta_2 \int_y \mu_{B_2}(y) y dy}{\beta_1 \int_y \mu_{B_1}(y) dy + \beta_2 \int_y \mu_{B_2}(y) dy}$$

cu $\beta_1 = 1 - \beta_2$ şi $\int_y \mu_{B_1}(y) dy = \int_y \mu_{B_2}(y) dy$.

$$y' = \frac{\beta_1 \int_y \mu_{B_1}(y) y dy}{\int_y \mu_{B_1}(y) dy} + \frac{\beta_2 \int_y \mu_{B_2}(y) y dy}{\int_y \mu_{B_2}(y) dy} = \beta_1 b_1 + \beta_2 b_2$$

Astfel, metoda de defuzificare COG în combinație cu operatorii de agregare, alții decât operatorul sumă va introduce neliniarități în funcția de ieşire a regulatorului. Metodele care vor fi prezentate în continuare sunt înrudite cu metoda COG, dar evită problemele anterior prezentate.

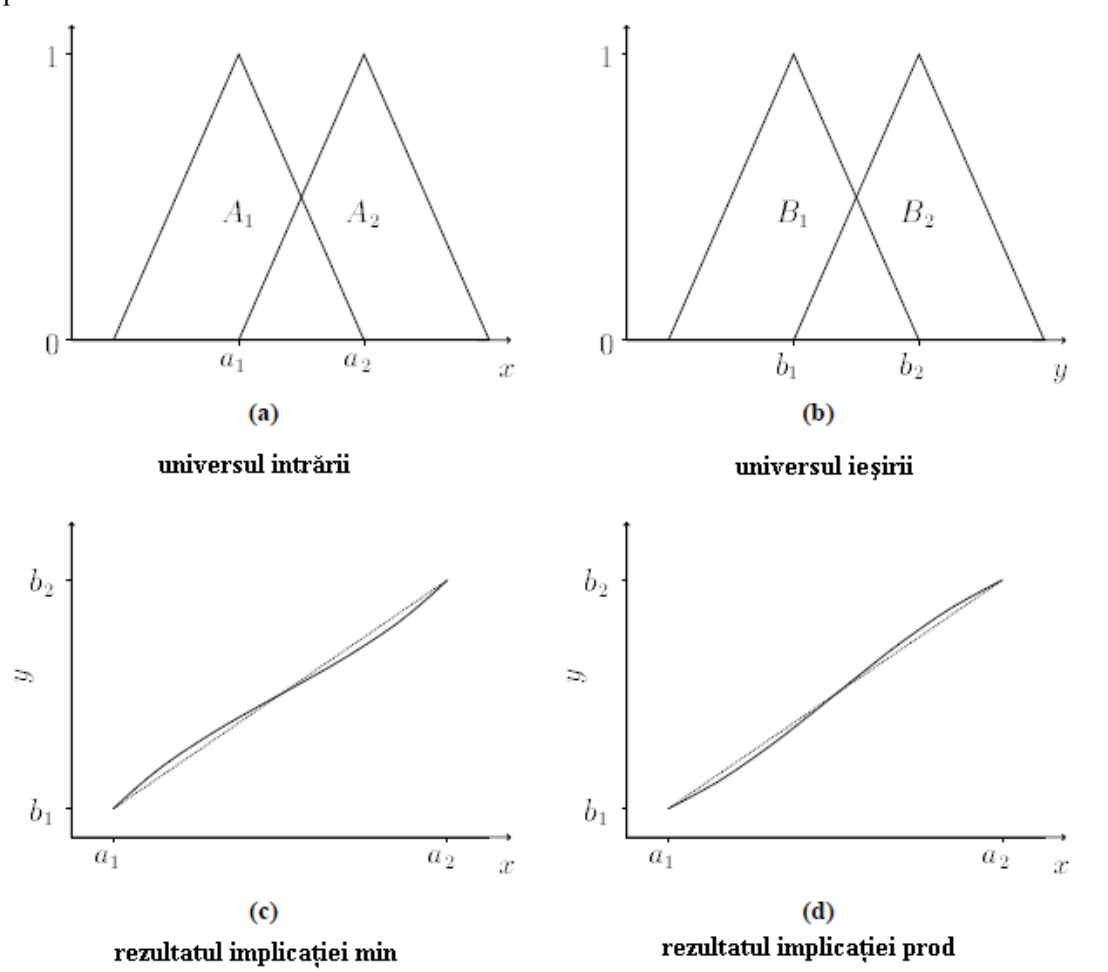


Fig.28. Funcția neliniară a ieşirii regulatorului în raport cu intrarea, când se utilizează metoda COG în combinație cu agregarea max. Liniile punctate reprezintă rezultatul interpolării liniare.

Metoda **fuzzy-mean (FM)** se aseamănă cu metoda COG discretizată în sensul că introduce o sumă ponderată. Diferența este că metoda COG discretizată şi metoda FM este că COG discretizează ieşirea fuzzy a regulatorului, în timp ce FM utilizează reprezentări numerice pentru mulțimile fuzzy din universul ieşirii regulatorului. Metoda fuzzy-mean este definită prin:

$$fm(B') = \frac{\sum_{j=1}^{N_B} \beta_j b_j}{\sum_{j=1}^{N_B} \beta_j}$$

unde N_B este numărul de mulțimi fuzzy definite pe universul de discuție al ieşirii regulatorului, β_j reprezintă gradul de îndeplinire pentru funcția de apartenență B_j ca rezultat al inferenței şi b_j exprimă reprezentarea numerică a lui B_j. Se presupune că valorile suportului b_j sunt cunoscute pentru fiecare mulțime de ieşire fuzzy B_j. Este posibil ca fiecare regulă să fie tratată separat, caz în care

$$fm(B') = \frac{\sum_{k=1}^{Nr} \beta_k b_k}{\sum_{k=1}^{Nr} \beta_k}$$

unde N_R este numărul de reguli fuzzy.

Utilizând metoda FM se combină practic metoda de agregare şi de defuzificare într-una singură. Din punct de vedere teoretic, rezultatul poate fi o mulțime fuzzy anormală, în contradicție cu teoria mulțimilor fuzzy. Rezultatul inferenței în practică nu este considerat o mulțime, ci o extindere (*a bag*, Yager,1994).

Din moment ce reprezentările numerice b_j pentru mulțimile fuzzy B_j pot fi precalculate, metoda de defuzificare FM este adecvată în cazul în care efortul de calcul în regulatorul fuzzy trebuie sa fie mic. Reprezentarea numerică b a unei mulțimi fuzzy B este în mod normal aleasă ca "centrul " acelei mulțimi:

$$b = dfz(B)$$

unde dfz este metoda de defuzificare, în cele mai multe cazuri metoda MOM.

9.2.4.2. Metode relative la înălțimea mulțimilor fuzzy

In literatura de specialitate, metodele de defuzificare utilizează în general metoda MOM descrisă în 8.1.2.3. Alte metode de defuzificare relative la înălțime sunt first-height (FHGT) sau first-maximum şi last-height sau last-maximum (LHGT). Aceste metode de defuzificare iau în considerare partea cea mai din stânga sau cea mai din dreapta a părții domeniului unde funcția de apartenență este egală cu înălțimea mulțimii fuzzy.

$$fhgt(B') = y' \quad \text{cu} \quad \mu_{B'}(y') = hgt(B') \quad \text{şi} \quad \forall y < y', \mu_{B'}(y) < \mu_{B'}(y')$$

$$lhgt(B') = y' \quad \text{cu} \quad \mu_{B'}(y') = hgt(B') \quad \text{şi} \quad \forall y > y', \mu_{B'}(y) < \mu_{B'}(y').$$

Utilizarea acestor metode depinde într-o mare măsură de aplicația de reglare avută în vedere.

Deşi metoda mean of maxima este utilizată în mod frecvent pentru defuzificare, aplicarea ei în reglarea fuzzy (dacă se utilizează implicații T) reduce gradul de „fuzificare" a regulatorului în multe cazuri. Dacă se utilizează MOM pentru defuzificarea ieşirii fuzzy a regulatorului, funcțiile de apartenență care descriu intrările pot fi alese drept mulțimi clasice, fără ca ieşirea numerică să fie afectată în vreun fel. Un exemplu simplu este prezentat în fig.29.

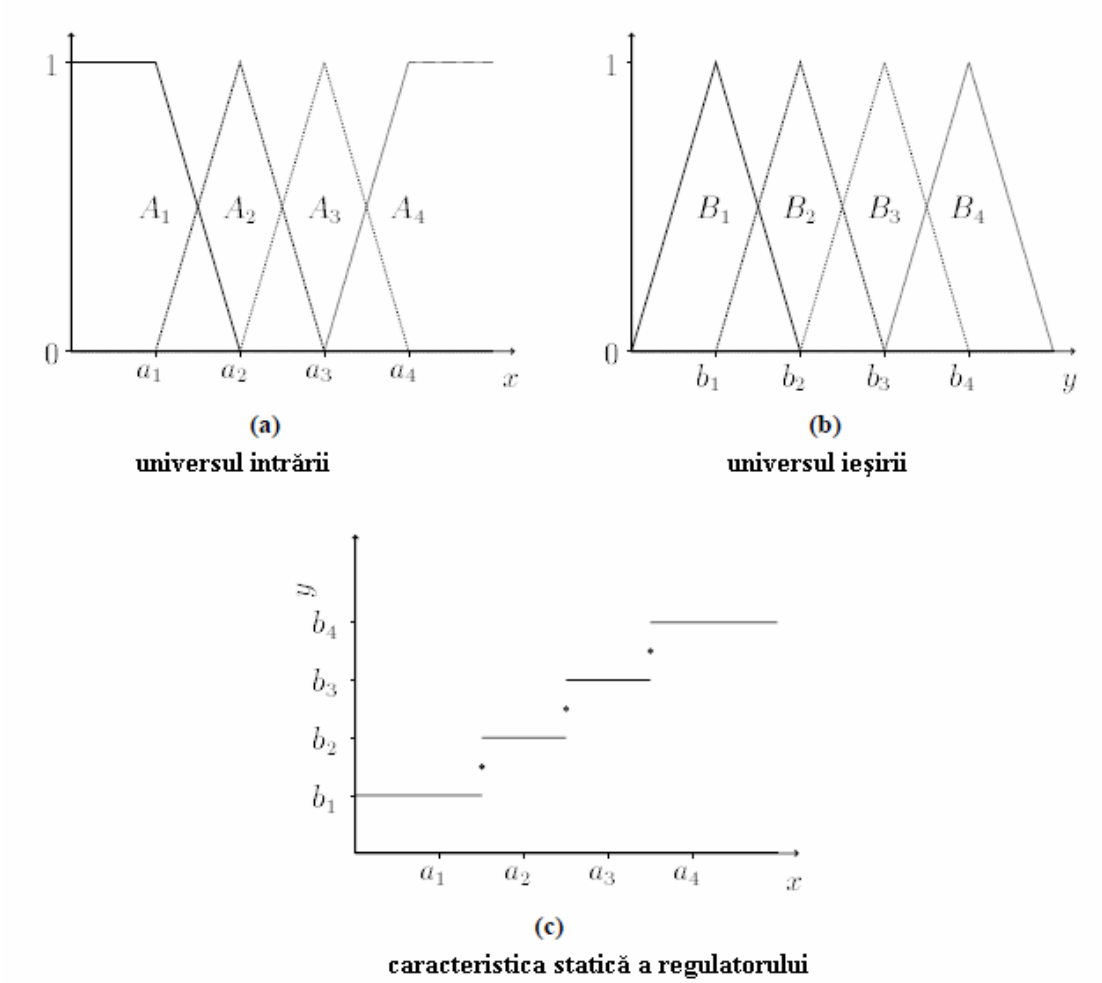


Fig.29. Exemplu de caracteristică statică a regulatorului în cazul defuzificării MOM. Regulatorul este bazat pe 4 reguli: dacă x este A_k atunci y este B_k, k=1,…,4.

Regulatorul fuzzy lucrează ca un releu multinivel datorită acestei metode MOM (Kickert şi Mamdami,1978). Se observă că există valori intermediare pentru ieşire acolo unde funcțiile de intrare se intersectează. Aceasta înseamnă că pentru acel proces nu era necesară utilizarea mulțimilor fuzzy. Există şi alte metode de defuzificare extinse ce pot fi găsite în literatura de specialitate. Aceste metode au ca principală trăsătură faptul că permit proiectantului regulatorului fuzzy să acordeze defuzificarea astfel încât să respecte anumite restricții.

In secţiunea 9.5 regulatorul fuzzy este privit ca o mapare intrare-ieşire şi este arătat că toate combinațiile de operatori logici şi metode de defuzificare (cu excepția unor combinații) vor genera neliniarități netriviale în maparea intrare-ieşire executată în sistemul fuzzy.

9.3 Reguli fuzzy de reglare

In reglarea fuzzy se disting două tipuri de reguli fuzzy: Mamdani şi Sugeno. Cele de tip Mamdami au fost discutate până acum în această lucrare. Regulile Sugeno se bazează pe un principiu diferit: consecinţele acestor reguli sunt funcţii (liniare) ale intrărilor regulatorului. Aceste două tipuri de reguli vor fi descrise în secţiunile următoare.

9.3.1 Regulile fuzzy Mamdani

Acest tip de reguli a fost utilizat în prima aplicaţie raportată de reglare fuzzy (Mamdani, 1974; Assilian, 1974; Mamdani şi Assilian, 1975) şi are forma generală:

r_k : **dacă** x_1 este A_1^k **şi ...**: : : **şi** x_{N_X} este $A_{N_X}^k$

atunci y_1 este B_1^k , : : : , y_{N_Y} este $B_{N_Y}^k$

Această formă a fost considerată până acum în lucrare. De exemplu, o regulă uzuală pentru un regulator este:

dacă eroarea este mare **şi** variaţia erorii este mică **atunci** comanda este puternică

Aceasta se aseamănă cu comportamentul unui regulator PD discret a cărui comandă se calculează în funcţie de eroare (diferenţa dintre referinţă şi ieşirea procesului) şi de viteza de variaţie a acesteia. Un asemenea regulator fuzzy va putea fi completat cu alte variabile faţă de un regulator clasic PD. De exemplu, regulatorul poate fi îmbunătăţit dacă se va utiliza în calculul comenzii variaţia ieşirii procesului în locul variaţiei erorii sau dacă semnalul de referinţă va fi clasificat astfel încât să surprindă şi caracteristicile neliniare ale procesului. Spre deosebire de un regulator clasic PID, un regulator fuzzy poate lucra după ambele principii de reglare.

In mod normal se utilizează operatorul *min* pentru conjuncţie şi implicaţie, operatorul *max* pentru agregare şi compoziţia *max-min* (Mamdani and Assilian, 1975). Această descriere corespunde metodei de inferenţă *max-min* (secţiunea 8.2.3.1). Mamdani şi colaboratorii săi au utilizat acest tip de regulă fuzzy şi au fost primii care au raportat o aplicaţie a logicii fuzzy în reglare (de aici şi numele de regulă Mamdani). Regulile fuzzy Mamdani au consecinţe de tip propoziţii fuzzy, iar implicaţia este reprezentată de conjuncţie (implicaţie de tip T).

Utilizarea unui număr limitat de funcţii de apartenenţă pentru ieşirea regulatorului fuzzy va conduce la restricţii impuse hipersuprafeţei de reglare, mai ales dacă intrările sunt normale şi formează o partiţie fuzzy, iar intrările sunt numerice. Dacă se consideră un regulator fuzzy cu o singură intrare şi o singură ieşire, funcţia regulatorului este limitată la o interpolare a punctelor caracteristice (figura 30). Această limitare este cauzată de utilizarea unui singure etichete lingvistice definită pentru ieşirea regulatorului la aplicarea regulii fuzzy. De exemplu, când intrarea numerică x' este chiar în centrul mulţimii fuzzy A_i, funcţia de apartenenţă asociată $\mu_{A_i}(x') = 1$ şi există o singură regulă activă. In figura 30 sunt date mulţimile fuzzy pentru intrarea (a) şi ieşirea regulatorului (b). In figura (c) linia continuă reprezintă rezultatul utilizării metodei de inferenţă *sum-prod* şi metoda de defuzificare *fuzzy-mean*. Regulatorul fuzzy conţine 4 reguli:

Dacă x este $A_1 / A_2 / A_3 / A_4$ **atunci** y este $B_1 / B_2 / B_3 / B_4$. Funcţia regulatorului este restricţionată la interpolarea între un număr fix de perechi (a_i, b_i). Cu linie punctată este reprezentat rezultatul utilizării consecinţelor ponderate multiple (de exemplu „**dacă** x este A_2 **atunci** y este $B_1(50\%)$, y este $B_2(50\%)$”).

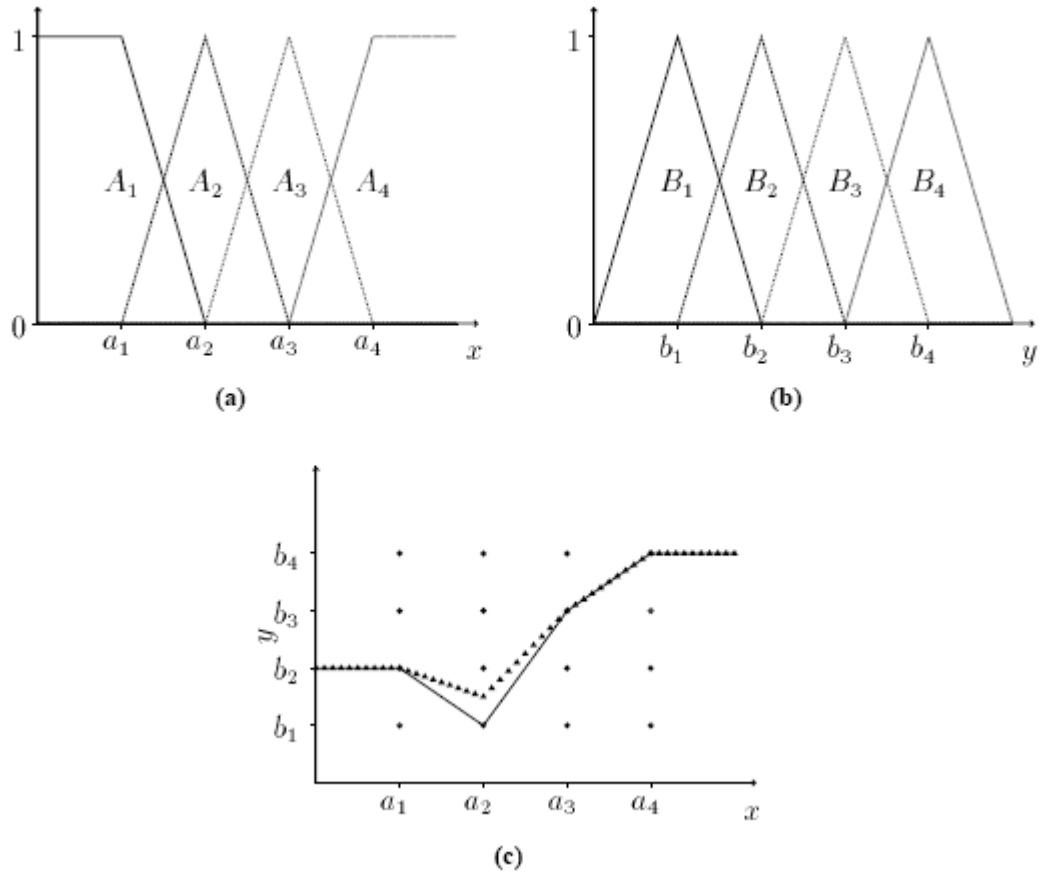


Figura 30. Limitările regulatorului fuzzy ce utilizează regulile Mamdani.

Această limitare a hipersuprafeţei de reglare poate fi redusă prin definirea mai multor mulţimi fuzzy pe universul intrării. Dezavantajul acestei metode este însă creşterea numărului de reguli şi a complexităţii sistemului (baza de reguli devine mai greu de înţeles). O altă posibilitate de reducere este utilizarea în regulile fuzzy a unor consecinţe multiplu ponderate (Jager ş. A., 1991), de exemplu:

dacă eroarea este mare **şi** variaţia erorii este medie
atunci variaţia comenzii este medie (80%), variaţia comenzii este mare (20%)

Această metodă permite obţinerea oricărei hipersuprafeţe de reglare, este un truc pentru a putea folosi reprezentarea lingvistică fuzzy de tip „mare”, „medie” conform cu cunoştinţele operatorului.

9.3.2 Regulile fuzzy Sugeno

Un alt tip de reguli fuzzy este denumit regulile Sugeno, datorită introducerii acestui tip de reguli de către Takagi şi Sugeno (1983), reguli dezvoltate mai târziu de Sugeno şi colaboratorii săi. In literatura de specialitate le regăsim şi ca regulile Takagi-Sugeno su regulile TS. Forma generală a unei reguli TS este:

r_k : **dacă** x_1 este A_1^k **şi ...**: : : **şi** x_{N_X} este $A_{N_X}^k$
atunci $y_1 = f_{1,k}(x_1,...,x_{N_x})$, : : : , $y_{N_Y} = f_{N_Y,k}(x_1,...,x_{N_X})$

In această formă generală se observă că regulile fuzzy au consecinţe care sunt funcţii ale intrărilor regulatorului x_i. In acest paragraf va fi considerată o singură ieşire pentru regulator. Sugeno şi colaboratorii săi au utilizat funcţii liniare pentru consecinţele regulilor fuzzy:

r_k : **dacă** x_1 este X_1^k **şi ...**: : : **şi** x_{N_X} este $X_{N_X}^k$ **atunci** $y = b_{0,k} + \sum_{i=1}^{N_X} b_{i,k} x_i$

unde $b_{i,k}$ şi $b_{0,k}$ sunt parametri constanţi. Un exemplu de regulă fuzzy TS asociată unui regulator este:

dacă e este mare **şi** Δe este mică **atunci** $c = 4e + 2\Delta e$

unde c este comanda regulatorului şi e şi Δe reprezintă eroarea şi valoarea ei iniţială. Primele utilizări cu succes ale acestui tip de reguli se regăsesc în literatura de specialitate la conducerea unei maşini, Sugeno şi Murakami (1984), Sugeno şi Nishida (1985) şi Sugeno şi Murakami (1985). Regulile Sugeno pot fi simplificate şi mai mult dacă se foloseşte $b_{i,k} = 0, \quad i = 1,..., N_X$, adică consecinţele sunt constante numerice $b_{0,k}$. Astfel setul de reguli Sugeno simplificate poate fi privit ca un set de regulatoare locale. In acest caz se poate folosi pentru defuzificare o sumă ponderată, ca o interpolare între ieşirile regulatoarelor locale, poate cea mai simplă metodă de defuzificare.
In cazul consecinţelor funcţii liniare se poate scrie:

$$y' = \frac{\sum_{k=1}^{N_r} \beta_k \left\{ b_{0,k} + \sum_{i=1}^{N_X} b_{i,k} x_i \right\}}{\sum_{k=1}^{N_r} \beta_k} = \frac{\sum_{k=1}^{N_r} \beta_k b_{0,k} + \sum_{i=1}^{N_X} \left\{ \sum_{k=1}^{N_r} \beta_k b_{i,k} x_i \right\}}{\sum_{k=1}^{N_r} \beta_k} = b_0' + \sum_{i=1}^{N_X} b_i' x_i ,$$

unde β_k este valoarea suportului regulii r_k şi:

$$b_0' = \frac{\sum_{k=1}^{N_r} \beta_k b_{0,k}}{\sum_{k=1}^{N_r} \beta_k}, \quad b_i' = \frac{\sum_{k=1}^{N_r} \beta_k b_{i,k}}{\sum_{k=1}^{N_r} \beta_k}, \text{ pentru } i = 1,..., N_X .$$

Cele de mai sus sugerează că un regulator fuzzy bazat pe reguli TS poate fi văzut ca un regulator în regim supervizor, care modifică parametrii unui regulator liniar. Un exemplu în acest sens este un regulator fuzzy cu reguli TS la care consecinţele alcătuiesc algoritmul PID. Un astfel de sistem este constituit dintr-un supervizor fuzzy cu reguli TS şi un regulator convenţional PID, situat la un nivel ierarhic inferior, ai cărui parametrii sunt modificaţi după regulile TS.

Din alt punct de vedere, o bază de reguli TS poate fi privită ca un set de regulatoare locale, fiecare cu setul propriu de parametrii de reglare. Acest mecanism este identic cu cel al adaptării programate („gain scheduling”) deşi nu este recunoscut ca tehnică fuzzy: definirea unor parametrii de reglare diferiţi în funcţie de diversele combinaţii/situaţii la intrare. Aplicând mulţimi fuzzy, mecanismul de inferenţă şi defuzificare rezultă aşa numita adaptare programată fuzzy („fuzzy gain scheduling”) unde tranziţia de la un set de parametrii de reglare la altul se face lin. Există însă şi posibilitatea apariţiei unor tranziţii discontinui de la un set de parametrii de reglare la altul (soluţia găsită în literatura de specialitate este denumită transfer „bumpless”, Åström şi Wittenmark, 1984).

Când se utilizează reguli TS, comanda regulatorului fuzzy este de fapt o sumă ponderată de funcţii ale intrărilor regulatorului. Din cauza acestei sume ponderate trebuie realizată o interpolare între consecinţele rezultate, interpolare ce prezintă dezavantajul că rezultatul obţinut nu poate fi intuit, mai ales când regulile TS sunt proiectate în mod grafic. Un exemplu tipic este dat de regulile următoare:

r_1: dacă x este A_1 atunci l_1: $y = c_1 x + d_1$
r_2: dacă x este A_2 atunci l_2: $y = c_2 x + d_2$

unde parametrii c_1, c_2, d_1 şi d_2 sunt aleşi astfel încât liniile l_1 şi l_2 se intersectează între centrele funcţiilor de apartenenţă ale etichetelor lingvistice A_1 şi A_2 definite pentru intrarea x a regulatorului. Presupunând că mulţimile fuzzy definite pentru intrarea x a regulatorului formează o partiţie fuzzy, comanda regulatorului are valori între funcţiile l_1 şi l_2. In figura 31 este dat un exemplu de comandă a regulatorului fuzzy cu reguli TS.

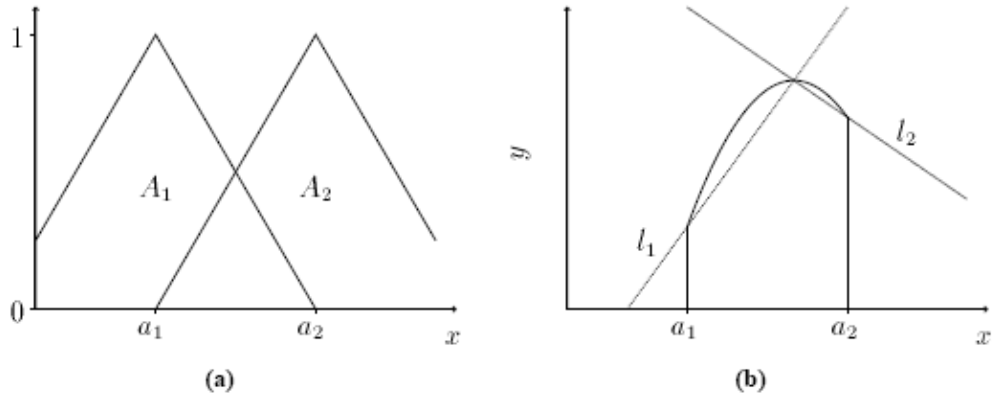


Figura 31. Comanda regulatorului (b) ca o funcţie a intrărilor utilizând reguli TS cu consecinţe funcţii liniare; a_1 şi a_2 sunt centrele mulţimilor fuzzy A_1 şi A_2 (a).

In mod evident consecinţele liniare nu descriu aproximările locale ale funcţiei regulatorului, mai precis cele referitoare la derivata funcţiei regulatorului. Babuška, Jager and Verbruggen (1994) susţin că aceste probleme sunt cauzate de factorul de scalare (bias) care provine din parametrii constanţi $b_{0,k} \neq 0$.

9.4 Reglarea fuzzy liniară

Prin reglare fuzzy liniară se înţelege reglarea proceselor (neliniare) bazate pe structuri de modele liniare cu parametri ce sunt determinaţi de un sistem fuzzy. Un asemenea sistem fuzzy poate fi privit ca un sistem supervizor (deci situat la un nivel ierarhic superior), diferenţa faţă de sistemele ierarhice apare însă la baza de timp pentru operarea în regim supervizor care este aceeaşi cu cea a reglării convenţionale a procesului. In cele ce urmează se vor numi „parametri fuzzy” acei parametri ai regulatorului calculaţi de un sistem fuzzy. Modelele şi regulatoarele bazate pe aceste modele fuzzy liniare au o structură liniară şi unul sau mai mulţi parametri fuzzy. Parametrii fuzzy ai modelelor liniare depind de variabilele de intrare ale procesului (mărimi de execuţie şi perturbaţii) dar şi de variabilele de ieşire ale acestuia, astfel încât modelul complet este de fapt un model neliniar al procesului.

9.4.1 Modele liniare fuzzy

In această secţiune va fi descrisă ideea de model liniar fuzzy. Modelul descris este chiar cel utilizat în proiectarea regulatorului cu metoda alocării polilor. Se consideră următorul proces neliniar:

$$\tau(y)\,\dot{y}(t) + y(t) = K(y)u(t - T_m)$$

unde:

- constanta de timp a procesului este $\tau(y) = \alpha_\tau + \beta_\tau |y|$;
- factorul de proporţionalitate este $K(y) = \alpha_K e^{-\beta_K |y|}$;
- timpul mort T_m.

Modelul fuzzy liniar pentru acest proces este descris de următorul model ARX (AutoRegresive model with eXternal input) cu parametri variabili:

$$y[k+1] = \tilde{a}_m[k]y[k] + \tilde{b}_m[k]u[k+d]$$

unde parametrii $\tilde{a}_m$ şi $\tilde{b}_m$ sunt descrişi de reguli fuzzy sau, de exemplu un tabel de căutare într-o bază de reguli fuzzy în combinaţie cu un algoritm de interpolare. Se poate utiliza orice metodă de modelare pentru obţinerea relaţiilor fuzzy dintre parametrii modelului şi semnalele de care aceştia depind. Acest model poate fi descris printr-un număr de reguli Sugeno:

dacă modulul lui $y[k]$ este în jur de y_j **atunci** $y[k+1] = a_j[k]y[k] + b_j[k]u[k+d]$

unde consecinţele liniare sunt modelate cu tehnici convenţionale de modelare. Fiecare regulă reprezintă un submodel şi premisa fuzzy a regulilor „în jur de y_j” în combinaţie cu CRI furnizează interpolarea între submodele. Pentru a obţine o interpolare cu neliniarităţi netriviale se recomandă folosirea metodei de inferenţă „sum-prod”. Pentru a obţine un model fuzzy liniar se utilizează metode de modelare standard pentru determinarea modelelor locale. De exemplu, factorul de proporţionalitate în regim

staţionar la un sistem neliniar poate să depindă de ieşirea din proces, această relaţie putând fi determinată prin identificarea sistemului cu diverse semnale de probă. In cele ce urmează vor fi detaliate modalităţile de proiectare a regulatoarelor bazate pe modele fuzzy.

9.4.2 Regulatoare liniare fuzzy

In figura 32 este prezentată schema regulatorului fuzzy proiectat cu metoda alocării polilor. Polinoamele R, S şi T sunt ale regulatorului. Polinoamele A şi B reprezintă modelul I-E al procesului sub forma unei funcţii de transfer. Regulatorul are componente feedback şi feedforward:

$$R(q)c[k] = T(q)i(k) - S(q)y(k)$$

unde $R(q)$, $S(q)$ şi $T(q)$ sunt polinoamele discrete ale regulatorului cu:

$c[k]$ - comanda către proces;

$i[k]$ - semnalul de referinţă;

$y[k]$ - ieşirea procesului.

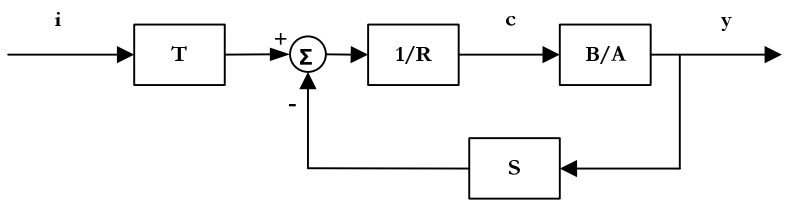


Fig. 32. Schema regulatorului fuzzy cu alocarea polilor.

Modelul procesului $G(z)$ este:

$$G(z) = \frac{B(z)}{A(z)} = \frac{b}{z-a}$$

unde a şi b sunt parametrii fuzzy ai polinoamelor $A(z)$ şi respectiv $B(z)$. Aceşti parametri sunt determinaţi de o regulă fuzzy la fiecare perioadă de eşantionare. Funcţia de transfer dorită pentru bucla închisă conform metodei alocării polilor se alege o funcţie de transfer de ordinul 2:

$$G_m(z) = \frac{B_m(z)}{A_m(z)} = \frac{(1+p_1+p_2)z^{-1}}{1+p_1z^{-1}+p_2z^{-2}}$$

unde:

$$p_1 = -2\exp(-\xi\omega T_s)\cos(\omega T_s\sqrt{1-\xi^2})$$

$$p_2 = \exp(-2\xi\omega T_s)$$

cu:

ξ - factorul de amortizare dorit;

ω - pulsaţia naturală dorită;

T_s - perioada de eşantionare.

Parametrii regulatorului ξ şi ω pot fi aleşi pentru a defini tipul dorit de răspuns al procesului. Astfel rezultă durata regimului tranzitoriu T_{Tr} şi suprareglajul σ la modificarea treaptă a referinţei:

$$T_{Tr} = \frac{4}{\xi\omega}\sec;$$

$$\sigma = \exp\left(-\frac{\pi\xi\omega}{\omega\sqrt{1-\xi^2}}\right)\%.$$

Polinoamele regulatorului provin din:

$$\frac{BT}{AR+BS} = \frac{B_m}{A_m}.$$

Când $R(z)$ este ales astfel încât să minimizeze eroarea staţionară la intrare treaptă e_{st}, se calculează următoarele polinoame:

$$R(z) = 1 - z^{-1};$$

$$S(z) = \frac{p_1 + a + 1}{b} + \frac{p_2 - a}{b} z^{-1};$$

$$T(z) = \frac{1 + p_1 + p_2}{b}.$$

In următorul paragraf se vor prezenta câteva rezultate obţinute cu regulatorul fuzzy cu metoda alocării polilor.

9.4.3 Experimente cu regulator fuzzy cu alocarea polilor

Relaţiile fuzzy dintre $K(y)$, $\tau(y)$ şi $|y|$ sunt determinate fiecare de cinci reguli fuzzy simple, în condiţiile în care sunt cinci funcţii de apartenenţă pe fiecare pe fiecare univers de discuţie pentru $K(y)$, $\tau(y)$ şi $|y|$. Tabelul 9.2. prezintă centrele funcţiilor de apartenenţă pentru cele 5 reguli.

Tabelul 9.2

$\|y\|$	0	0.25	0.5	0.75	1
$K(y)$	2	1.75	1.2	0.95	0.75
$\tau(y)$	0,5	0.75	1	1.25	1.5

Modelul dorit pentru comportarea în buclă închisă are ca parametrii:

$\xi = 0.95$;

$\omega = 3.5$;

$T_s = 0.1$;

$T_m = 0.1$.

Procesul neliniar utilizat în acest experiment are modelul:

$$\underbrace{\left(\frac{1}{2}+|y|\right)}_{\tau(y)}\dot{y}(t)+y(t)=\underbrace{2e^{-|y|}}_{K(y)}u(t-T_m)$$

unde $T_m = 0.1$. Perioada de eşantionare utilizată este $T_s = 0.1$. Rezultatele obţinute sunt prezentate în figura 33.

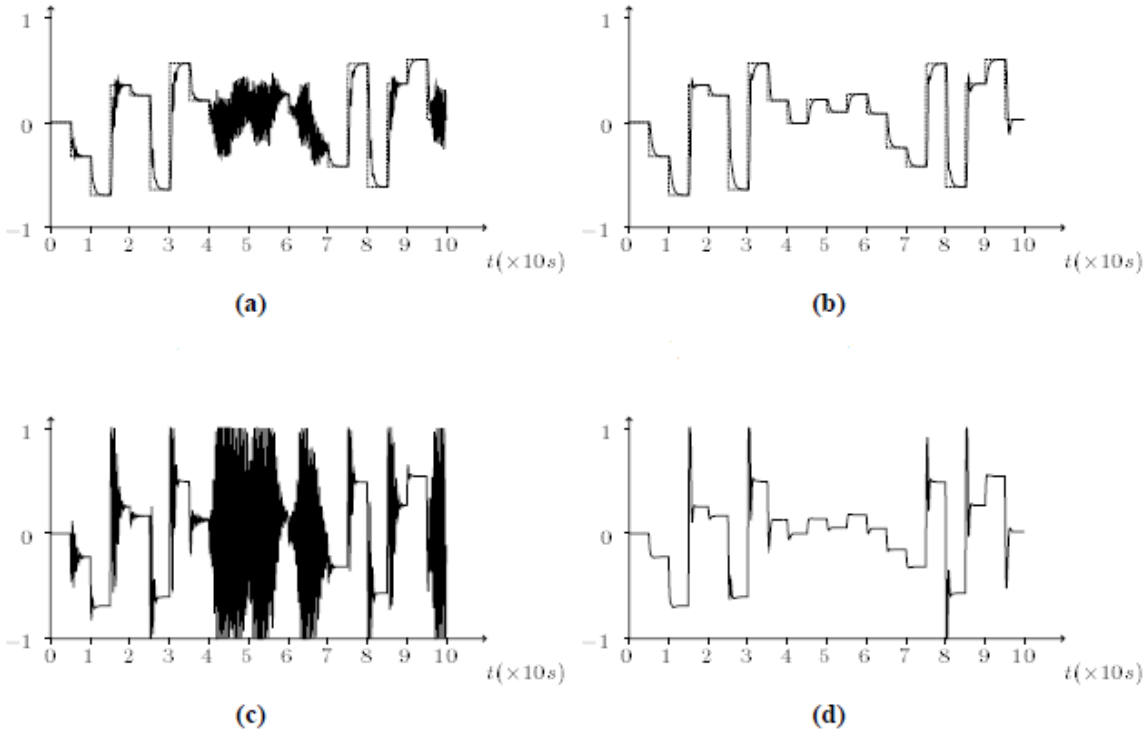


Fig.33. Rezultatele experimentelor cu regulator liniar şi fuzzy cu metoda alocării polilor (MAP). Coloana stânga arată rezultatele pentru cazul procesului liniar (τ=1, K=1), regulator liniar MAP. Coloana dreaptă arată rezultatele regulatorului fuzzy cu MAP. (a), (b) – referinţa şi ieşirea din proces, (c), (d) – comanda regulatorului.

Figura 33 arată că se îmbunătăţesc performanţele sistemului când se utilizează regulator liniar fuzzy cu MAP. In acest exemplu e acelaşi rezultat care s-ar fi obţinut utilizând regulile Takagi-Sugeno. Prima abordare realizează interpolare între parametrii modelului continuu, a doua abordare realizează interpolarea între diferitele valori ale ieşirii regulatorului discret.

Obs. Modelarea proceselor cu structuri de modele liniare cu parametri fuzzy reprezintă o metodă de modelare a proceselor neliniare. Aceste modele cu parametrii fuzzy care depind de semnale cunoscute a priori pot compensa neliniarităţile procesului ce trebuie modelat.

9.5 Regulatorul fuzzy ca o funcţie de tip intrare-ieşire

În această secţiune este descris regulatorul fuzzy ca o funcţie de tip intrare-ieşire. Dacă privim regulatorul fuzzy ca o reprezentare statică a intrărilor în raport cu ieşirile, în cele mai multe cazuri suprafaţa de reglare este neliniară. Funcţia care descrie această reprezentare se notează cu

$$y = f(x)$$

Ieşirea numerică y' este defuzificarea mulţimii fuzzy de ieşire B', care la rândul ei este rezultatul aplicării regulii de compoziţie a inferenţei CRI:

$$y' = f(x') = dfz(B') = dfz(A' \circ R).$$

Cu siguranţă acest tip de funcţie este dificil de analizat, însă funcţia statică pe care o căutăm poate fi privită ca rezultatul interpolării multiple în spaţiul multidimensional, evident atunci când sunt îndeplinite anumite condiţii. Aceste condiţii sunt date în secţiunea următoare. În secţiunea 9.5.2. se arată cum un regulator fuzzy poate emula un regulator liniar.

9.5.1 Sistemul fuzzy - estimator universal

In introducerea anterioară s-a afirmat că regulatorul fuzzy poate fi privit ca o funcţie statică de tip intrare-ieşire $y = f(x)$. Regulile fuzzy definesc această funcţie $f(x)$ pentru că ele reprezintă cunoştinţele pe baza cărora s-a construit modelul fuzzy. Printre alţii, Kosko (1992) a demonstrat că sistemele fuzzy sunt estimatoare universale. Adică un sistem fuzzy poate aproxima orice alt sistem:

$$\forall x \in X, \quad |F(x) - f(x)| < \varepsilon$$

unde $F(x)$ este funcţia care trebuie aproximată iar ε poate fi ales oricât de mic. Dacă sistemul fuzzy este privit ca o aproximare discretă a lui $F(x)$ şi este cunoscut faptul că pentru obţinerea unei aproximări continue este nevoie de o interpolare obţinută de obicei prin inferenţa fuzzy, în mod evident dacă se utilizează mai multe reguli se obţine o estimare mai precisă (în mod similar cu eşantionarea unei funcţii continue cu o anumită perioadă, cu cât perioada de eşantionare este mai mică, cu atât precizia de aproximare este mai mare).

Sistemul fuzzy poate fi privit ca o interpolare între puncte, fiecare definit de o regulă fuzzy. Trebuie observat faptul că interpolarea se realizează doar între punctele caracteristice, fiecare reprezentat de una din regulile fuzzy, dacă se îndeplinesc următoarele condiţii:

- mulţimile fuzzy definite pentru intrări şi ieşiri formează partiţii fuzzy descrise de $\forall x \in X, \quad \sum_{j=1}^{N_A} \mu_{A_j}(x) = 1.$, cu $A_j \neq \Phi$ şi $A_j \neq X$;
- funcţiile de apartenenţă sunt convexe şi normale;
- baza de reguli este completă.

Când aceste condiţii sunt îndeplinite, sunt cel mult 2^{N_x} reguli active, rezultând o interpolare între consecinţele acestor reguli fuzzy active în spaţiul N_x- dimensional. In

secţiunea următoare se descrie pe larg acest fenomen. Funcţia rezultată poate să fie neliniară ca rezultat al regulilor fuzzy, dar şi al alegerii operatorilor fuzzy, a funcţiilor de apartenenţă, etc. Analiza efectelor acestor parametri este prezentată în paragraful 9.6.

9.5.2 Regulator liniar $\subset$ regulator fuzzy

S-a arătat că orice regulator liniar poate fi descris de un regulator fuzzy: reglarea fuzzy include şi reglarea liniară sau reglarea liniară este o submulţime a reglării fuzzy. Atunci când se proiectează un regulator fuzzy şi se aplică diferitele alegeri pentru forma funcţiilor de apartenenţă, operatorii logici fuzzy şi scalarea intrărilor şi ieşirilor fuzzy un regulator fuzzy poate emula un regulator liniar. In continuare vor fi considerate doar intrările numerice. In cazul unui regulator liniar, funcţia asociată este considerată o ecuaţie algebrică de tipul:

$$y = c^T x + d = \sum_{i=1}^{N_x} c_i x_i + d ,$$

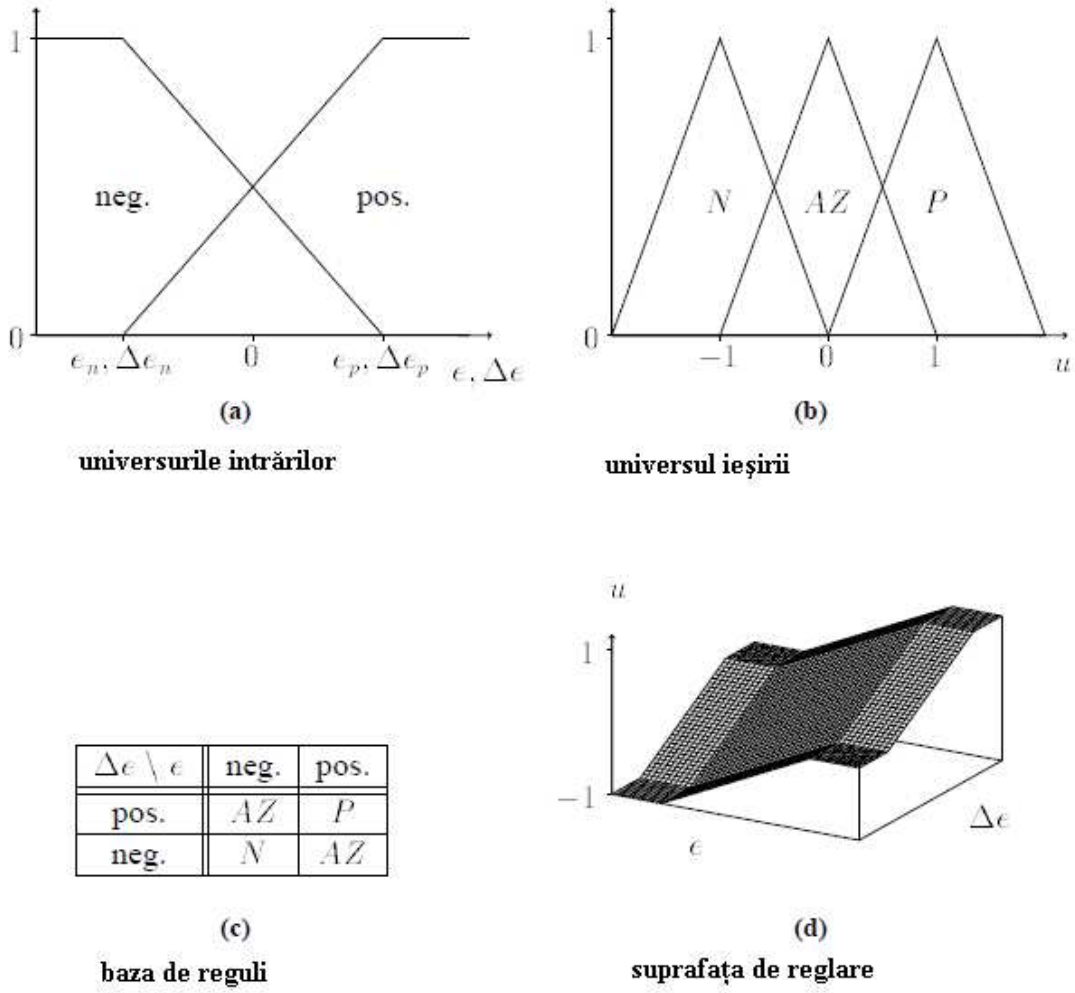
unde d este un offset (factor de compensare). Regulatorul fuzzy cu funcţia $y = f(x)$ poate emula un regulator liniar atunci când sunt îndeplinite următoarele criterii:

L-1 funcţiile de apartenenţă definite pe universul de discuţie asociat intrărilor să fie de formă triunghiulară şi normale;
L-2 mulţimile fuzzy pentru fiecare intrare să formeze o partiţie fuzzy (suma funcţiilor de apartenenţă să fie 1);
L-3 baza de reguli fuzzy să fie completă;
L-4 Să fie utilizată o normă T pentru funcţia implicaţie (implicaţie-T);
L-5 operatorul conjuncţie în premisele regulilor fuzzy să fie operatorul produs;
L-6 pentru agregare şi pentru conectorul *sau* (dacă se utilizează) să se folosească operatorul *sum* (*mărginit*) (uniune conform lui Łukasiewicz);
L-7 consecinţele defuzificate (reprezentări numerice constante) ale fiecărei reguli fuzzy să fie alese conform ecuaţiei $y = c^T x + d = \sum_{i=1}^{N_x} c_i x_i + d$;
L-8 să fie utilizată metoda fuzzy-mean (evident cu utilizarea operatorului de agregare cerut în L-6).

In majoritatea cazurilor găsite în literatura de specialitate sunt îndeplinite condiţiile L-1 până la L-4. Criteriile L-1 până la L-4 presupun existenţa unei reguli fuzzy pentru fiecare combinaţie de intrări. Cerinţa L-8 este şi ea îndeplinită de majoritatea regulatoarelor fuzzy. Aceasta conduce la ideea că principalele diferenţe dintre regulatoarele fuzzy standard şi cele liniare fuzzy sunt date de criteriile L-5 până la L-7. Utilizarea operatorilor sumă şi produs în locul operatorilor *max* şi *min* este necesară pentru emularea cât mai fidelă a caracterului liniar, obţinându-se astfel o interpolare liniară.
Cel mai important criteriu care trebuie îndeplinit este L-7, pentru că datorită celorlalte criterii ieşirea unui regulator fuzzy cu N_X intrări se calculează ca o sumă ponderată a 2^{N_X} puncte într-un spaţiu N_X - dimensional. Consecinţele numerice ale celor mai active 2^{N_X} reguli fuzzy determină existenţa sau lipsa unei relaţii liniare prin interpolarea acestor

puncte (din hiperplan). Dacă există acest hiperplan, funcţia de intrare-ieşire a sistemului fuzzy e echivalentă cu $y = c^T x + d = \sum_{i=1}^{N_x} c_i x_i + d$.

Un sistem fuzzy (regulator sau model) poate emula orice sistem liniar atâta vrem cât acesta are o descriere statică funcţională. Numărul minim necesar de reguli este 2^{N_X} pentru N_X intrări. In figura 34 se arată cum se poate emula un regulator liniar PD cu ajutorul unui regulator fuzzy cu 4 reguli. Dacă se înlocuieşte comanda cu derivata comenzii în consecinţele regulilor, se poate emula şi un regulator liniar PI.



$\Delta e \setminus e$	neg.	pos.
pos.	AZ	P
neg.	N	AZ

(c)
baza de reguli

Fig.34. Regulator fuzzy emulator de regulator PD. Eroarea e şi prima diferenţă Δe sunt intrările în regulator (a); comanda u este ieşirea regulatorului (b). Baza de reguli este prezentată în (c) şi suprafaţa de reglare în (d). Se observă că suprafaţa de reglare este liniară pentru anumite intervale ale intrărilor.

In aplicaţiile de reglare fuzzy emularea unui regulator liniar poate fi punctul de plecare în proiectarea unui regulator fuzzy, urmând rafinarea acestuia prin adăugarea sau modificarea regulilor fuzzy. In cazul regulilor Sugeno care au consecinţele descrise de

funcţii există o altă posibilitate de emulare a unui regulator liniar. De altfel este o metodă foarte simplă: toate regulile trebuie să aibă ecuaţii de tipul $y = c^T x + d = \sum_{i=1}^{N_x} c_i x_i + d$.

Datorită însumării ponderate a rezultatelor regulilor individuale, comanda regulatorului are forma impusă de L-7:

$$y = \frac{\sum_{k=1}^{N_r} \beta_k (c_k^T x + d)}{\sum_{k=1}^{N_r} \beta_k} = \frac{(c^T x + d) \sum_{k=1}^{N_r} \beta_k}{\sum_{k=1}^{N_r} \beta_k} = c^T x + d \ .$$

Aşa cum se observă, nu există restricţii asupra funcţiilor de apartenenţă sau a normelor T utilizate în inferenţa regulii sau pentru conectorul *şi*. Există o singură restricţie în sensul că trebuie să fie cel puţin o regulă activă al un moment dat, pentru că în cazul regulilor Sugeno fiecare consecinţă poate fi un algoritm liniar.

Deşi abordarea de tip emulator pentru un regulator liniar pare trivială şi lipsită de utilitate, este un punct de plecare în proiectarea un regulator fuzzy neliniar.

Lucrarea 1
Simularea sistemelor cu logică fuzzy în Matlab: prezentarea Toolbox-ului Fuzzy Logic

A. Obiective

1) Deprinderea folosirii mediului de simulare Matlab pentru dezvoltarea de aplicaţii care folosesc sisteme cu logică fuzzy.
2) Deprinderea modalităţilor de analiză a funcţionării sistemelor cu logică fuzzy oferite de mediul de simulare Matlab.

B. Concepte teoretice necesare lucrului cu mediul de simulare Matlab pentru construirea şi simularea sistemelor cu logică fuzzy

Pentru a putea construi şi simula un sistem cu logică fuzzy în Matlab, este necesară cunoaşterea elementelor prin care se defineşte un sistem cu logică fuzzy: variabile de intrare şi de ieşire ale sistemului cu logică fuzzy; mulţimi fuzzy de intrare şi de ieşire; bază de reguli fuzzy şi a operaţiilor implicate de funcţionarea sistemului cu logică fuzzy: fuzzificare; inferenţă; agregare; defuzzificare. Aceste concepte au fost studiate în cadrul Cursului de Sisteme de Reglare Fuzzy.

C. Prezentarea Toolbox-ului Fuzzy Logic

Toolbox-ul Fuzzy Logic al mediului de simulare Matlab reprezintă mediul software de simulare a sistemelor cu logică fuzzy oferit de acest program extrem de popular în lumea ştiinţifică. Construirea unui sistem cu logică fuzzy în Matlab este simplă datorită interfeţei grafice prin care utilizatorul poate comunica cu mediul de simulare Matlab (componenta dedicată logicii fuzzy).
Această interfaţă permite construirea şi, în final, analiza grafică a sistemului cu logică fuzzy dorit, folosind cele 5 instrumente grafice care formează componenta dedicată logicii fuzzy a mediului Matlab:
1) **Editorul sistemului cu logică fuzzy**, numit FIS Editor (=*Fuzzy Inference System Editor*);
2) **Editorul funcţiilor de apartenenţă**, adică, al mulţimilor fuzzy peste universurile discursurilor variabilelor de intrare şi de ieşire ale sistemului cu logică fuzzy, numit *Membership Function Editor*;
3) **Editorul regulilor fuzzy**, care vor forma baza de reguli a sistemului cu logică fuzzy, numit *Rule Editor*;
4) **Fereastra de vizualizare a regulilor**, în care se poate observa gradul de activare al fiecărei reguli şi rezultatul (mulţimea fuzzy de ieşire) dat de fiecare regulă a bazei de reguli, pentru o anumită valoare tranşantă a intrării stabilite de către utilizator; această fereastră apare sub numele *Rule Viewer*;
5) **Fereastra de vizualizare a suprafeţei ieşire-intrare** a sistemului cu logică fuzzy, care arată grafic dependenţa ieşirii tranşante a sistemului cu logică fuzzy de intrările tranşante ale sistemului cu logică fuzzy, determinate de funcţionarea sistemului cu logică fuzzy; această dependenţă este cunoscută în general sub numele de "suprafaţă de reglare a sistemului", iar fereastra apare sub numele *Surface Viewer*.

Cele cinci instrumente grafice sunt legate dinamic între ele, adică, o modificare a sistemului cu logică fuzzy făcută într-una din ferestrele de editare va putea afecta ceea ce se vede în celelalte ferestre. La un moment dat, pentru un sistem cu logică fuzzy, putem avea oricâte din cele cinci ferestre deschise simultan.
Cele cinci instrumente grafice ale componentei dedicate sistemelor cu logică fuzzy din Matlab sunt prezentate simbolic în diagrama-bloc din Figura 1.

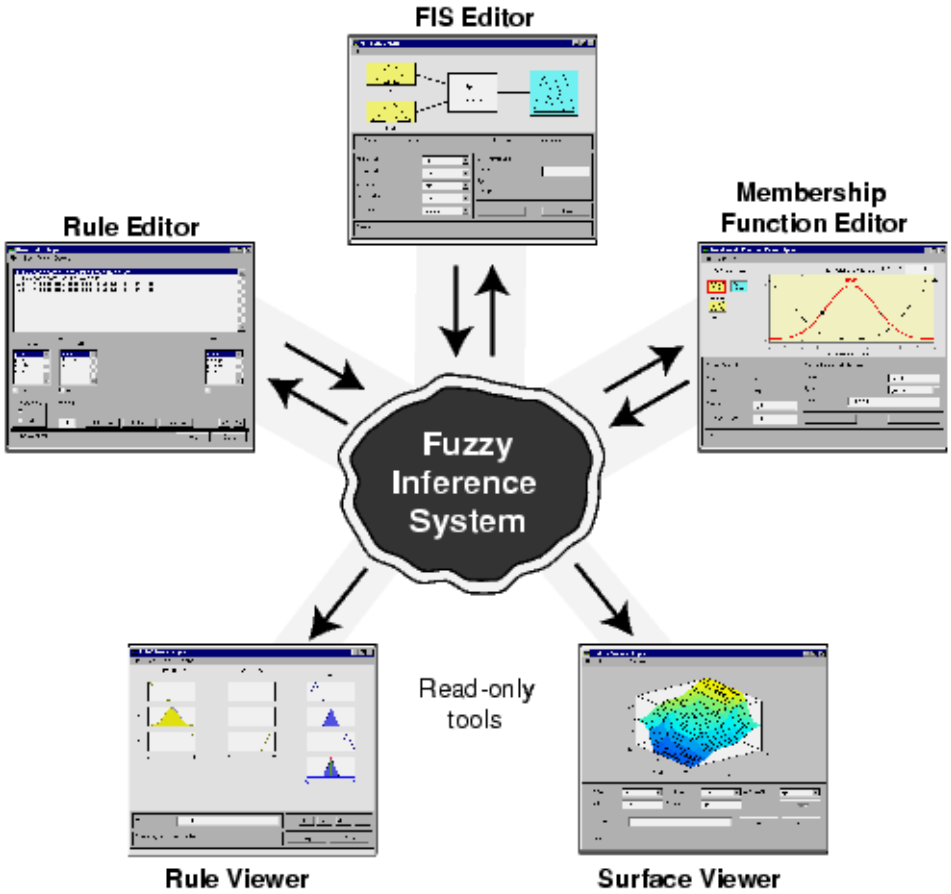


Figura 1. Cele cinci instrumente de construire şi analiză a unui sistem cu logică fuzzy şi legătura lor cu sistemul cu logică fuzzy

Pentru construirea unui sistem cu logică fuzzy în Matlab, primul pas după lansarea în execuţie a programului Matlab.exe este scrierea de la tastatură, la prompterul programului, a comenzii:

> **fuzzy**

urmată de apăsarea tastei Enter. Această comandă lansează în execuţie editorul sistemului cu logică fuzzy (FIS Editor); va apărea pe ecran fereastra grafică a editorului de sisteme cu logică fuzzy din Figura 2.

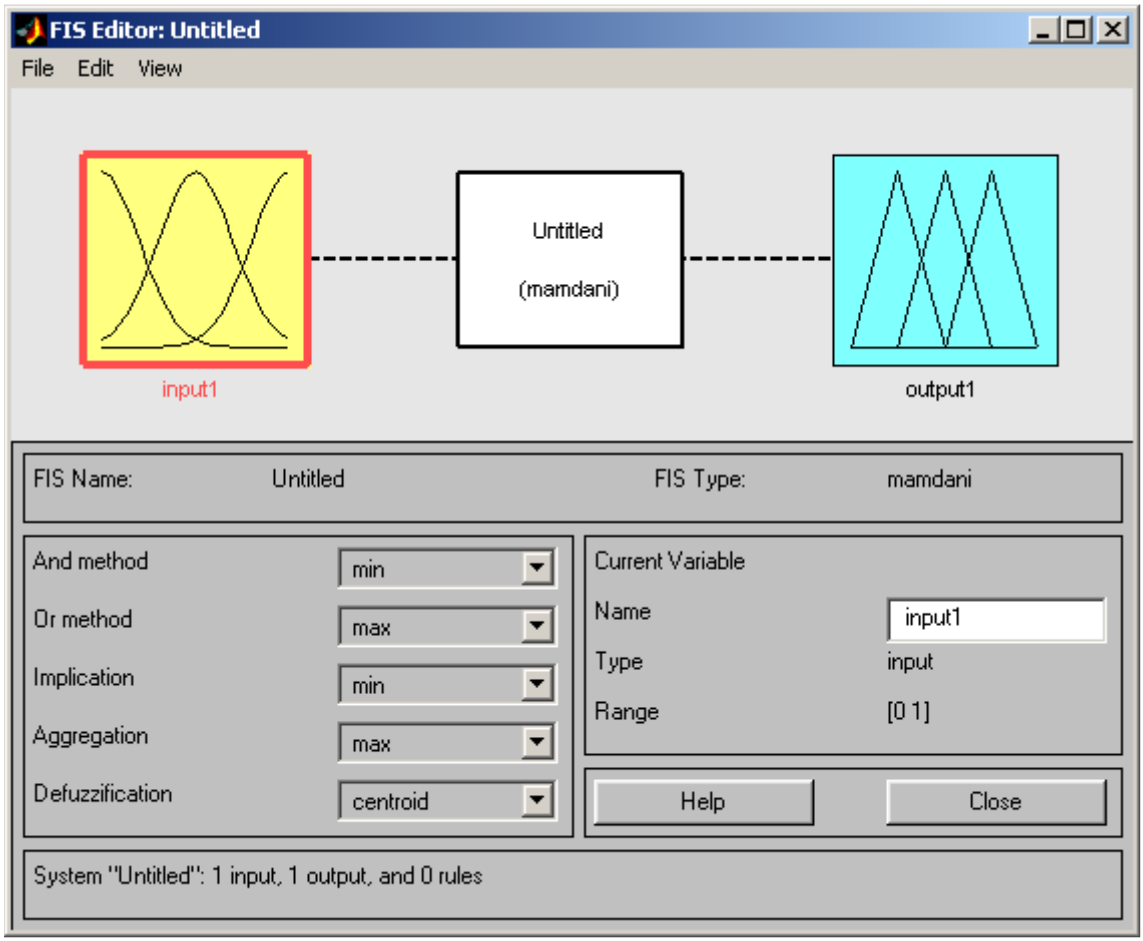


Figura 2. Fereastra editorului de sisteme cu logică fuzzy.

C1. Editorul sistemului cu logică fuzzy

Editorul sistemului cu logică fuzzy permite definirea informaţiilor generale despre sistemul cu logică fuzzy pe care dorim să-l construim. In diagrama simplă din partea de sus a ferestrei, de sub bara de meniuri, sunt afişate:

- în stânga, un dreptunghi în care apar generic trei funcţii de apartenenţă de formă Gauss, şi sub el, numele variabilei de intrare; în Figura 2, există (implicit) o singură variabilă de intrare în sistemul cu logică fuzzy, numită input1;
- în dreapta, un dreptunghi în care apar generic trei funcţii de apartenenţă de formă triunghiulară, şi sub el, numele variabilei de ieşire; în Figura 2, există (implicit) o singură variabilă de ieşire din sistemul cu logică fuzzy, numită output1;
- în centru, un dreptunghi în care apare numele sistemului cu logică fuzzy şi tipul de inferenţă folosit de către sistemul cu logică fuzzy; implicit, mecanismul de inferenţă folosit este Mamdani, iar numele sistemului cu logică fuzzy este Untitled.

Observaţie: Funcţiile de apartenenţă desenate în această fază a construcţiei sistemului cu logică fuzzy în dreptunghiurile din stânga şi din dreapta din diagramă sunt doar nişte simboluri, şi de fapt nu există nici pentru variabila de intrare, nici pentru cea de ieşire din sistemul cu logică fuzzy; momentan, nu este definită nici o mulţime fuzzy peste universurile discursurilor variabilelor de intrare input1 şi de ieşire output1.

Sub diagrama din partea de sus a ferestrei editorului sistemului cu logică fuzzy, în partea stângă a Figurii 2, este un sistem de meniuri de tip pop-up care ne permit să modificăm diferitele operaţii folosite în cadrul procesului de inferenţă:
- **And method** permite selecţia expresiei matematice folosite pentru operatorul AND din premisa regulii fuzzy: aceasta poate fi min sau prod (produs), şi implicit este min;
- **Or method** permite selecţia expresiei matematice folosite pentru operatorul OR, care este implicit operatorul care leagă intre ele regulile din baza de reguli fuzzy: aceasta poate fi max sau probor (sau probabilistic), şi implicit este max;
- **Implication** defineşte operaţia folosită pentru exprimarea relaţiei de implicaţie fuzzy dintre premisa şi consecinţa fiecărei reguli: aceasta poate fi min sau prod, şi implicit este min;
- **Aggregation** defineşte operaţia folosită pentru combinarea mulţimilor fuzzy obţinute ca ieşire a fiecărei reguli fuzzy din baza de reguli, cunoscute sub numele de concluzii parţiale, pentru a se obţine mulţimea fuzzy care reprezintă răspunsul sistemului cu logică fuzzy la o valoare dată a intrării tranşante: aceasta poate fi max, sum (sumă algebrică) sau probor, şi implicit este max;
- **Defuzzification** defineşte metoda de defuzzificare folosită: aceasta poate fi: centroid, bisector (bisectoarea ariei), lom (largest of maximum), mom (middle of maximum), som (smallest of maximum);

Observaţie: $probor(a,b) = a + b - ab$.

Sub diagrama din partea de sus a ferestrei editorului sistemului cu logică fuzzy, în partea dreaptă a Figurii 2, se găseşte o zonă în care sunt afişate:
- numele variabilei (de intrare sau ieşire) selectate;
- tipul funcţiilor de apartenenţă asociate variabilei de intrare sau ieşire selectate;
- universul discuţiei variabilei de intrare sau ieşire selectate;

Implicit, nici o variabilă nu este selectată. Selecţia unei variabile de intrare sau ieşire a sistemului cu logică fuzzy se face printr-o simplă apăsare cu butonul stâng al mouse-ului, atunci când cursorul mouse-ului este poziţionat pe blocul din diagrama sistemului cu logică fuzzy corespunzător variabilei respective.
De exemplu, dacă vrem să selectăm variabila de intrare input1, ne poziţionăm pe dreptunghiul galben din stânga schemei-bloc din Figura 2, apăsăm pe butonul stâng al mouse-ului şi în acest moment dreptunghiul respectiv va fi marcat ca şi activ prin înconjurarea sa cu o linie roşie.
Simultan, în partea dreaptă jos a Figurii 2 apare numele variabilei: input1. Acest nume poate fi acum modificat prin ştergerea câmpului Name şi scrierea de la tastatură a noului nume pentru variabila de intrare.
Celelalte două câmpuri rămân în continuare nespecificate (ele au doar nişte valori implicite), deoarece încă nu am specificat forma funcţiilor de apartenenţă şi universul discuţiei variabilelor sistemului cu logică fuzzy.
Bara de meniuri a editorului de sisteme cu logică fuzzy conţine următoarele comenzi:
- Comanda File – de aici, putem crea un nou sistem cu logică fuzzy cu mecanism de inferenţă de tip Mamdani (New Mamdani FIS); un nou sistem cu logică fuzzy cu inferenţă de tip Sugeno (New Sugeno FIS); putem importa un sistem cu logică fuzzy deja creat fie de pe hard disc (Import from disk), fie din spaţiul de lucru curent (workspace) (Import from workspace); putem

salva/exporta sistemul cu logică fuzzy creat/modificat fie pe hard disc (Import to disk), fie în spaţiul de lucru curent (Import to workspace).
- Comanda Edit – cu subcomenzile de: 1) refacerea situaţiei anterioare, Undo; 2)adăugarea unei noi variabile de intrare/ieşire în sistemul cu logică fuzzy, Add input/Add output; 3) ştergerea din sistemul cu logică fuzzy a variabilei de intrare sau ieşire selectate în momentul curent, Remove variable; 4) editarea mulţimilor fuzzy care acoperă universurile discursurilor variabilelor de intrare şi ieşire Membership functions; această comandă lansează în execuţie editorul funcţiilor de apartenenţă 5) editarea regulilor fuzzy din baza de reguli, Rules; această comandă lansează în execuţie editorul regulilor fuzzy Rules.
- Comanda View – cu subcomenzile de: 1) vizualizarea regulilor fuzzy, View rules;
această comandă lansează în execuţie fereastra de vizualizare a regulilor; 2) vizualizarea suprafeţei de control a sistemului cu logică fuzzy, View surface; această comandă lansează în execuţie fereastra de vizualizare a suprafeţei ieşire/intrare a sistemului cu logică fuzzy.
Rezumat: Paşii de urmat în editorul sistemelor cu logică fuzzy, FIS Editor, pentru crearea sistemului cu logică fuzzy dorit:
1) Dacă sistemul nostru are mai mult de o intrare, se dă din bara de meniuri comanda Edit->Add input; fiecare comandă de acest tip adaugă o nouă intrare sistemului cu logică fuzzy, denumită implicit: input2, input3,...
2) Se modifică numele variabilelor de intrare şi ieşire ale sistemului cu logică fuzzy, astfel încât să fie cât mai sugestive pentru aplicaţia dorită, selectând prin apăsarea butonului stâng al mouse-ului căsuţa din diagrama sistemului cu logică fuzzy corespunzătoare variabilei respective, iar apoi modificând conţinutul câmpului de editare care conţine numele variabilei, urmată de apăsarea tastei Enter.
3) Opţional, se pot modifica valorile operatorilor AND, OR, Implicaţie, Agregare, Defuzzificare din partea stângă a ferestrei editorului de sisteme cu logică fuzzy.
4) Se salvează sistemul cu logică fuzzy în spaţiul de lucru, cu comanda File->Export to workspace as... , cu numele dorit pentru sistemul creat. In acest moment, diagrama bloc a sistemului cu logică fuzzy se modifică în conformitate cu opţiunile introduse pentru sistem.
5) Următorul pas va fi definirea mulţimilor fuzzy care acoperă universurile discursurilor variabilelor de intrare şi ieşire. Acest lucru se realizează în editorul funcţiilor de apartenenţă, care se lansează din fereastra editorului FIS cu comanda:
Edit Membership Functions...

C2. Editorul funcţiilor de apartenenţă

Editorul funcţiilor de apartenenţă permite afişarea şi editarea tuturor funcţiilor de apartenenţă asociate cu toate variabilele de intrare şi de ieşire ale sistemului cu logică fuzzy. Fereastra grafică a editorului funcţiilor de apartenenţă este foarte asemănătoare cu cea a editorului sistemului cu logică fuzzy (Figura 3).

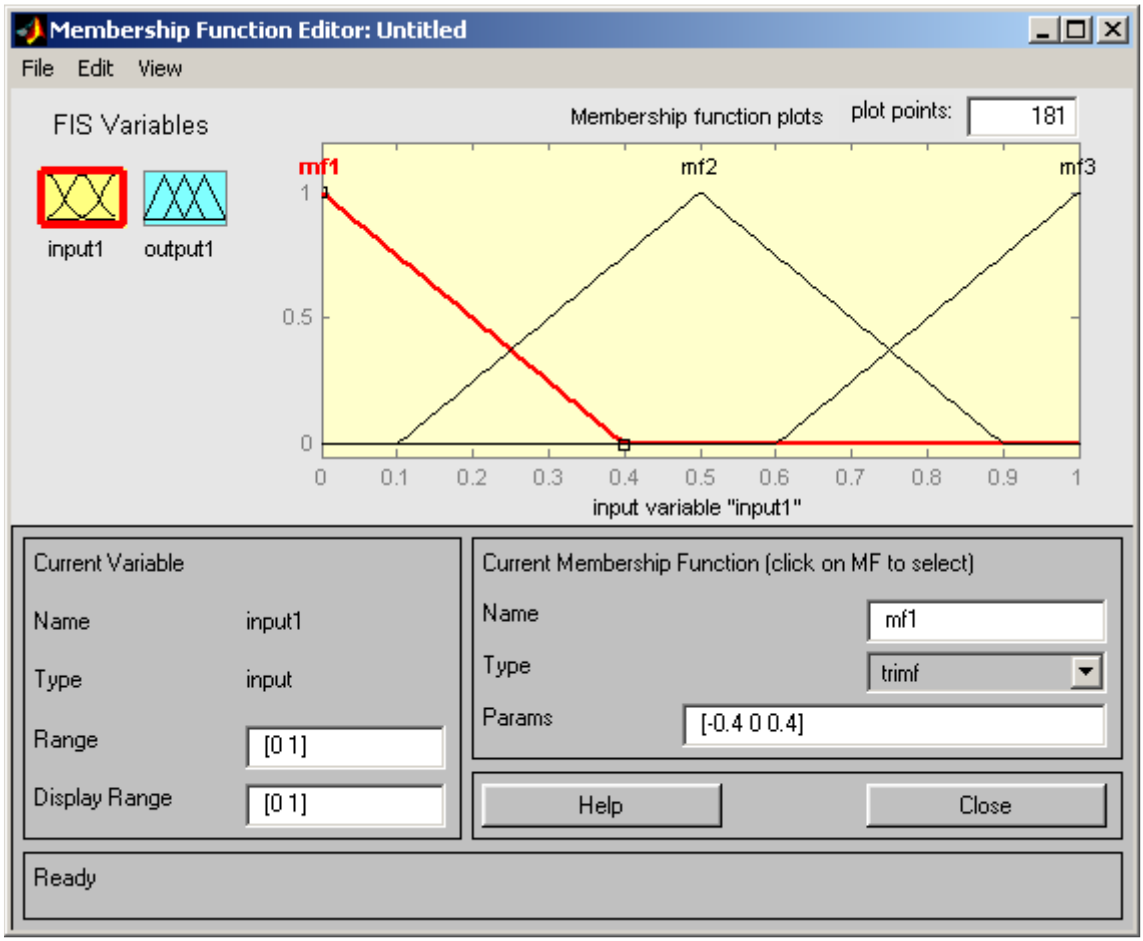


Figura 3. Fereastra editorului funcţiilor de apartenenţă.

La deschiderea editorului funcţiilor de apartenenţă pentru un sistem cu logică fuzzy nou creat, nu există încă nici o funcţie de apartenenţă asociată variabilelor de intrare şi ieşire definite anterior în editorul sistemului cu logică fuzzy.

In colţul din stânga sus al ferestrei editorului funcţiilor de apartenenţă există o "paletă a variabilelor", în care apar listate toate variabilele de intrare şi de ieşire ale sistemului, sub forma simbolului lor-bloc şi cu numele variabilei scris sub blocul respectiv. De exemplu, pentru un sistem cu o singură intrare şi o singură ieşire al căror nume a fost lăsat implicit, în listă apar variabilele: input1 şi output1.

Pentru a stabili funcţiile de apartenenţă asociate cu o variabilă din listă, trebuie selectată variabila printr-o apăsare cu butonul stâng al mouse-ului pe blocul care simbolizează variabila respectivă; în acest moment, atât blocul cat şi numele variabilei sunt "colorate" cu roşu, iar sub ecranul din partea dreaptă a ferestrei editorului, "Membership function plots", apare tipul şi numele variabilei respective.

Sub "paleta variabilelor", în partea stângă jos a ferestrei editorului, avem informaţii despre tipul şi numele variabilei selectate, în câmpurile: nume; tip (intrare sau ieşire); domeniu de variaţie (adică, universul discursului variabilei respective); domeniul valorilor afişate (care poate fi un subdomeniu al universului discursului). In câmpul domeniu de variaţie (range) vom specifica pentru fiecare variabilă universul discursului, sub forma unui interval de numere reale. Apăsând

apoi tasta Enter, domeniul este setat la cel dorit, şi implicit acesta va fi şi domeniul la care se setează câmpul domeniului valorilor afişate (display range).
Pentru definirea funcţiilor de apartenenţă ale variabilei selectate, se foloseşte componenta Edit a barei de meniuri a ferestrei editorului, din care se selectează comanda Add MFs (adaugă funcţii de apartenenţă). Va apărea o nouă fereastră pe ecran, ca în Figura 4, care permite:

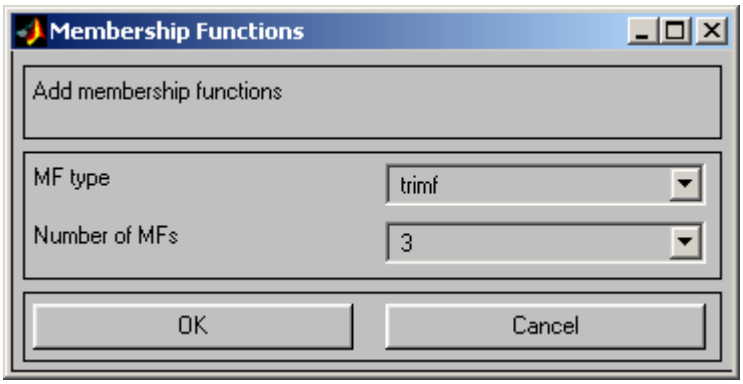


Figura 4. Fereastra de definire a funcţiilor de apartenenţă pentru o variabilă

1) selectarea numărului de funcţii de apartenenţă asociate variabilei selectate, între 1 şi 9 (implicit este 3);
2) selectarea tipului funcţiilor de apartenenţă asociate variabilei selectate: triunghi (trimf), trapez (trapmf), clopot (gbellmf), Gauss (gaussmf, gauss2mf), PI (pimf), combinaţii de două funcţii sigmoide (dsigmf, psigmf). Toate funcţiile de apartenenţă implicite în Matlab sunt simetrice în raport cu axa y, iar dacă dorim de exemplu să definim un triunghi dreptunghic pe extremele universului discuţiei, acesta se defineşte ca şi un triunghi isoscel al cărui suport se întinde şi în afara universului discuţiei.

Odată stabilit tipul şi numărul funcţiilor de apartenenţă pentru variabila selectată (prin apăsarea tastei OK a ferestrei din Figura 4), aceste funcţii apar afişate pe ecranul din partea dreaptă sus a ferestrei editorului. Fiecare funcţie de apartenenţă poate fi selectată prin apăsarea butonului stâng al mouse-ului.
In momentul când o funcţie de apartenenţă este selectată, ea va apărea desenată cu roşu, şi simultan în partea dreaptă jos a ferestrei editorului, cu titlul "Current membership function", apar afişate: numele, tipul (trimf, trapmf, etc) şi parametrii funcţiei de apartenenţă respective (de exemplu, pentru funcţiile triunghi şi trapez, pe care le vom folosi în acest îndrumar de laborator, adică, pentru funcţii liniare pe porţiuni, parametrii sunt valorile tranşante ale variabilei corespunzătoare punctelor de frângere din grafic). In acest moment, putem modifica: numele funcţiei de apartenenţă; tipul său; parametrii care descriu funcţia de apartenenţă.
Odată definite toate funcţiile de apartenenţă conform dorinţei proiectantului sistemului cu logică fuzzy, putem edita baza de reguli fuzzy a sistemului. Pentru aceasta, din meniul ferestrei editorului, se selectează comanda View, urmată de Edit rules....

C3. Editorul regulilor fuzzy

Editorul regulilor fuzzy permite editarea bazei de reguli fuzzy a sistemului cu logică fuzzy. Fereastra grafică a editorului regulilor este cea din Figura 5.

Inițial, câmpul de editare al ferestrei, de culoare albă, nu conține nici o regulă. Toate regulile din baza de reguli a sistemului cu logică fuzzy pe care dorim să-l proiectăm vor fi introduse utilizând butoanele din josul ferestrei. In meniul de sus al ferestrei editorului de reguli fuzzy apare un buton de Options ce permite selecția formatului de scriere a regulilor, Rule Format. Opțiunea selectată în acest câmp poate fi:

1) Format verbal (verbose), caz în care regulile se editează sub forma if...then; de exemplu, dacă atunci când variabila de intrare input1 aparține mulțimii fuzzy mf1, dorim ca variabila de ieşire output1 să aparțină mulțimii fuzzy mf1, regula va fi: If input1 is mf1 then output1 is mf1, şi exact sub această formă este editată în spațiul editorului de reguli. După terminarea scrierii tuturor regulilor din bază sub această formă, se transformî regulile în formatul din Figura 5.

2) Format simbolic (symbolic), caz în care regula de mai sus devine: (input1==mf1)=>(output1=mf1);

3) Format indexat (indexed), caz în care regula de mai sus devine: 1,1 (1):1;ultimul "1" indică folosirea operatorului AND în premisa regulii, atunci când este cazul.

Odată scrise într-un format oarecare, regulile pot fi vizualizate în orice format dorim.

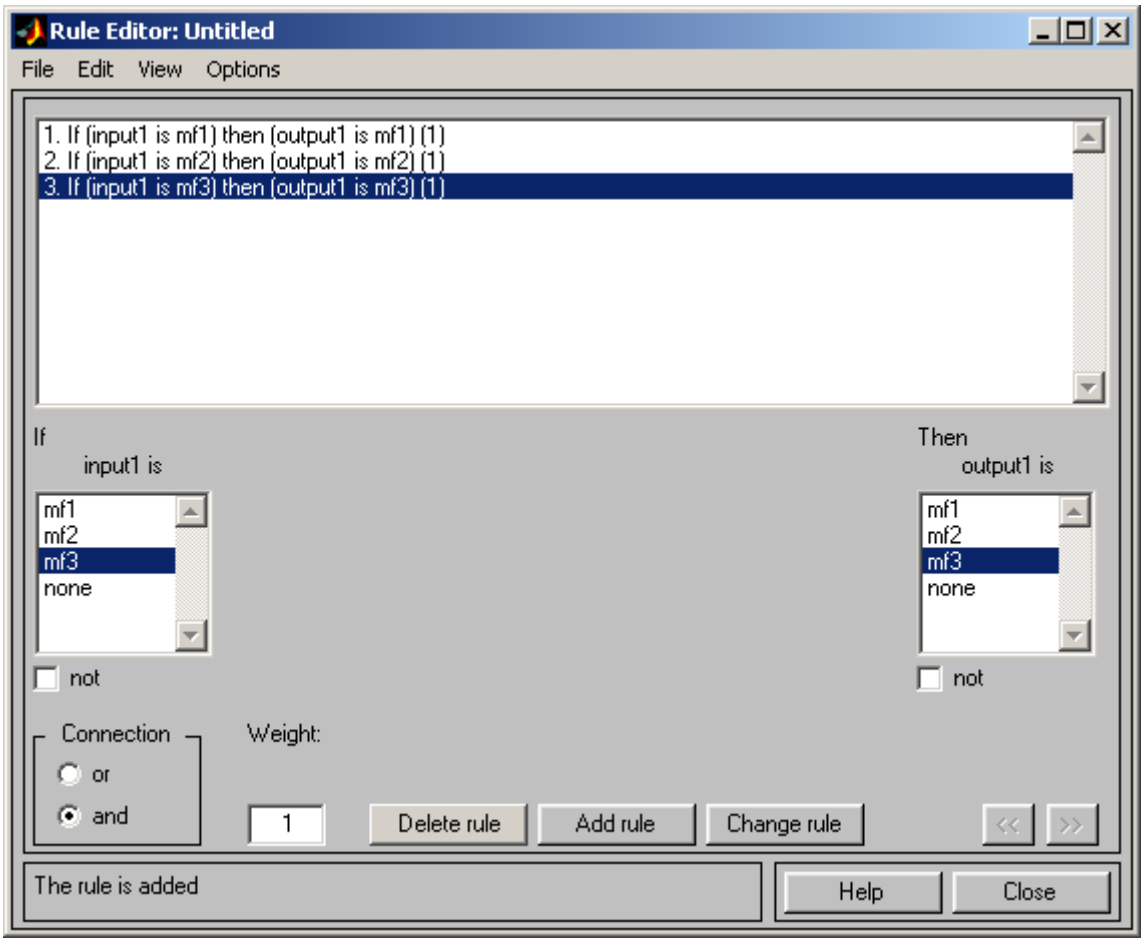


Figura 5. Fereastra editorului de reguli fuzzy.

C4.-C5. Ferestrele de vizualizare: a regulilor; a suprafeţei ieşire-intrare

Astfel, sistemul cu logică fuzzy este complet construit. Acum este indicată salvarea sa pe hard disc din orice fereastră de editare dorim, din meniul File->Export to disk as.... In continuare ne rămâne ca, folosind utilitarele grafice de vizualizare a regulilor fuzzy şi a suprafeţei de ieşire a sistemului cu logică fuzzy, să ne convingem că sistemul proiectat funcţionează aşa cum dorim.

D. Partea experimentală

D1. Indicaţii preliminare

Pentru partea experimentală, veţi folosi componenta dedicată simulării sistemelor cu logică fuzzy a mediului software Matlab, numită Fuzzy Logic Toolbox, scopul lucrării de laborator fiind tocmai învăţarea utilizării acestei componente în scopul proiectării sistemelor cu logică fuzzy. Sistemul cu logică fuzzy folosit pentru experimentare este cel de modelare a caracteristicii i-v a unei diode semiconductoare.
Experimentele presupun: descrierea sistemului cu logică fuzzy de modelare a caracteristicii $i-v$ a diodei folosind cele trei editoare grafice ale componentei fuzzy din Matlab, în fişierul Dioda.fis; simularea şi analiza sistemului cu logică fuzzy cu ajutorul celor două ferestre de vizualizare: a regulilor fuzzy şi a suprafeţei intrare-ieşire a sistemului cu logică fuzzy, care sunt instrumentele de depanare dedicate ale componentei fuzzy din Matlab; salvarea rezultatelor experimentale în fişiere de tip bitmap şi notarea valorilor măsurate prin simulare. La punctul E al lucrării, veţi interpreta experimentele făcute cu scopul de a răspunde la setul de întrebări formulate.

D2. Descrierea aplicaţiei

Una dintre aplicaţiile importante ale sistemelor cu logică fuzzy este folosirea lor ca aproximanţi de funcţii. Fiind dată o funcţie grafică $y = f(x)$, care poate fi reprezentată doar aproximativ sub formă analitică, s-a demonstrat în literatura de specialitate că se poate proiecta un sistem cu logică fuzzy având x ca intrare şi y ca ieşire, astfel încât între valoarea estimată a lui y, $\hat{y}$ şi $f(x)$ există o diferenţă oricât de mică dorim, $\left|\hat{y} - f(x)\right| \to 0$. Cu alte cuvinte, sistemul cu logică fuzzy cu intrarea x şi ieşirea y este un aproximant universal de funcţii grafice $y = f(x)$, adică aproximează $f(x)$ cu eroarea ε oricât de mică.
In aplicaţia prezentată în lucrare dorim să aproximăm regiunea directă a caracteristicii statice curent-tensiune $(i-v)$ a unei diode semiconductoare, pe baza curbei $i-v$ a componentei determinate experimental (sau luate din catalog). Ca intrare, sistemul cu logică fuzzy va avea valorile tensiunii pe diodă la polarizare directă, $v = v_{AK}$, exprimate în mV, în domeniul [0 ... 860 mV]. Ca ieşire, sistemul cu logică fuzzy trebuie să furnizeze valoarea curentului anodic prin diodă, $i = i_A$, exprimat în mA, în domeniul [0 ... 450 mA].

Sistemul cu logică fuzzy obţinut va caracteriza complet comportamentul electric al diodei în regiunea directă pentru $v \in [0...860\,mV]$, ignorând variaţia caracteristicii $i - v$ cu temperatura, umiditatea şi abaterile de proces tehnologic de fabricaţie a diodei. Deci, sistemul cu logică fuzzy va putea fi folosit ca un model electric al diodei.

D2.1. Principiul de modelare fuzzy a caracteristicii $i - v$ a diodei

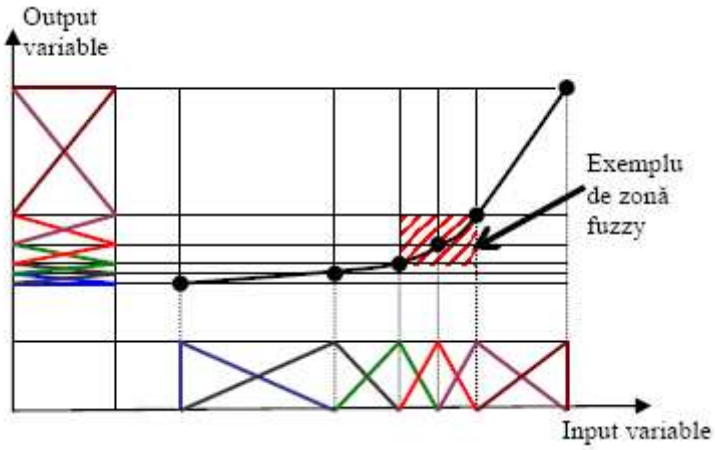


Figura 6. Acoperirea caracteristicii neliniare $i - v$ a diodei prin zone fuzzy.

Principiul modelării caracteristicii $i - v$ a diodei folosit în această lucrare se bazează pe acoperirea graficului neliniar din Figura 6 prin "zone fuzzy". Fiecare "zonă fuzzy" este descrisă printr-o mulţime fuzzy peste universul discuţiei variabilei de intrare v, notată MFV_K, o mulţime fuzzy corespunzătoare peste universul discuţiei variabilei de ieşire i, notată MFI_K şi o regulă fuzzy care indică faptul că între cele două mulţimi fuzzy MFV_K şi MFI_K există o relaţie fuzzy care descrie "zona fuzzy": Rk: Dacă v este MFV_K atunci i este MFI_K.

Pe porţiunile de grafic cu neliniaritate pronunţată sunt necesare mai multe mulţimi fuzzy de intrare şi respectiv de ieşire pentru o descriere precisă a graficului. In orice zonă aproximativ liniară a graficului, o pereche de mulţimi fuzzy este suficientă pentru o descriere completă a graficului.

D2.2. Sistemul cu logică fuzzy pentru modelarea caracteristicii $i - v$ a diodei semiconductoare

Componentele sistemului cu logică fuzzy folosit în lucrare pentru modelarea caracteristicii $i - v$ a diodei semiconductoare sunt:

- mulţimile fuzzy definite peste universul discuţiei variabilei de intrare, v, notate prin MFV_K;
- mulţimile fuzzy definite peste universul discuţiei variabilei de ieşire, i, notate prin MFI_K;

baza de reguli fuzzy, conţinând regulile Rk de forma:

Rk: Dacă v este MFV_K atunci i este MFI_K.

Intrarea tranşantă a sistemului cu logică fuzzy este valoarea instantanee a tensiunii pe diodă la polarizarea directă, $v \in [0...860\,mV]$. Ieşirea tranşantă a sistemului cu logică fuzzy va fi curentul i prin diodă, $i \in [0...450mA]$.

Prin realizarea operațiilor de: fuzzificare; inferență; defuzzificare, operații care prelucrează valoarea tranșantă de intrare v a sistemului cu logică fuzzy în conformitate cu descrierea sistemului cu logică fuzzy dată de componentele sale, se obține valoarea tranșantă a ieșirii i; perechile de valori tranșante (v, i) obținute, reprezentate grafic în sistemul cartezian, aproximează cu eroare mică caracteristica $i - v$ a diodei semiconductoare reprezentată în Figura 6.

Structura sistemului cu logică fuzzy, cu intrarea și ieșirea tranșante, componentele și fluxul operațiilor sale, este reprezentată în Figura 7.

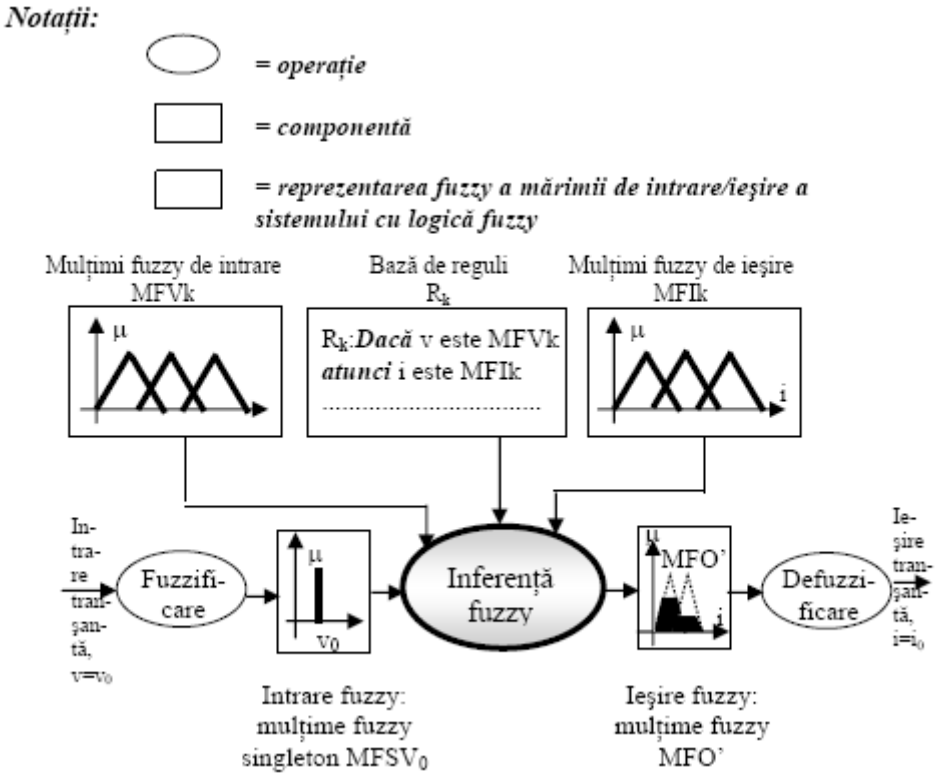


Figura 7. Structura sistemului cu logică fuzzy pentru aproximarea caracteristicii $i - v$ a diodei semiconductoare.

In această lucrare de laborator vom analiza numai operația de inferență, care poate fi considerată operația esențială pentru funcționarea unui sistem cu logică fuzzy. Totuși, pentru înțelegerea sa, o scurtă prezentare a celorlalte două operații este necesară:

- Operația de fuzzificare transformă mărimea tranșantă de intrare într-o mulțime fuzzy singleton; astfel, dacă la momentul considerat, valoarea tranșantă a intrării v este $v_0 \in [0...860mV]$, $v = v_0$ v=v0, atunci mulțimea fuzzy singleton de intrare rezultată în urma fuzzificării, notată prin $MFSI_0$, este definită prin funcția de apartenență:

$$\mu_{MSFI_0} = \begin{cases} 1, & v = v_0 \\ 0, & v \neq v_0 \end{cases}$$, reprezentată în Figura 8.

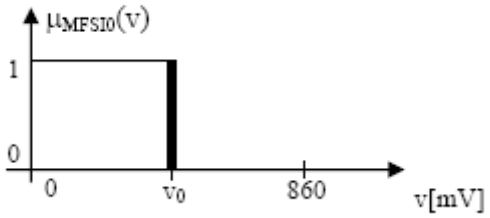


Figura 8. Funcţia de apartenenţă a mulţimii fuzzy singleton $MFSI_0$.

- Operaţia de defuzzificare selectează, folosind pentru aceasta un anumit criteriu, o valoare $i = i_0$ situată pe suportul mulţimii fuzzy de ieşire rezultate în urma inferenţei, notată prin MFO', ca valoare caracteristică ("prototip") a mulţimii fuzzy de ieşire; $i_0 \in [0...450mA]$.

Exemple de astfel de proceduri de selecţie pot fi: valoarea i_0 corespunzătoare centrului de greutate al lui MFO'; media coordonatelor i' cu proprietatea că $\mu_{MFO'}(i') = \max_i \mu_{MFO'}(i)$; cea mai mică valoare i_0 cu proprietatea că $\mu_{MFO'}(i_0) = \max_i \mu_{MFO'}(i)$ etc.

D2.2.1. Definirea mulţimilor fuzzy pentru variabila de intrare v

Universul discuţiei variabilei de intrare v poate fi citit de pe caracteristica $i - v$ din Figura 6, şi este intervalul de numere reale: [0 ... 860 mV].

Experimental se constată că o aproximare suficient de bună, în condiţiile păstrării complexităţii sistemului cu logică fuzzy în limite scăzute, se obţine pentru o acoperire a caracteristicii $i - v$ prin 9 "zone fuzzy". Aceasta conduce la o acoperire a universului discuţiei variabilei v prin 9 mulţimi fuzzy de formă triunghiulară; pentru simplitatea notaţiei, cele 9 mulţimi fuzzy sunt denumite: MFV1, MFV2, ..., MFV9. (MFV este o notaţie prescurtată pentru Mulţime Fuzzy pentru tensiunea V; indicele numeric corespunde "zonei fuzzy" respective). Cele 9 mulţimi fuzzy formează o partiţie fuzzy, iar coordonatele suporturilor şi valorilor lor mediane pot fi citite în Figura 9.

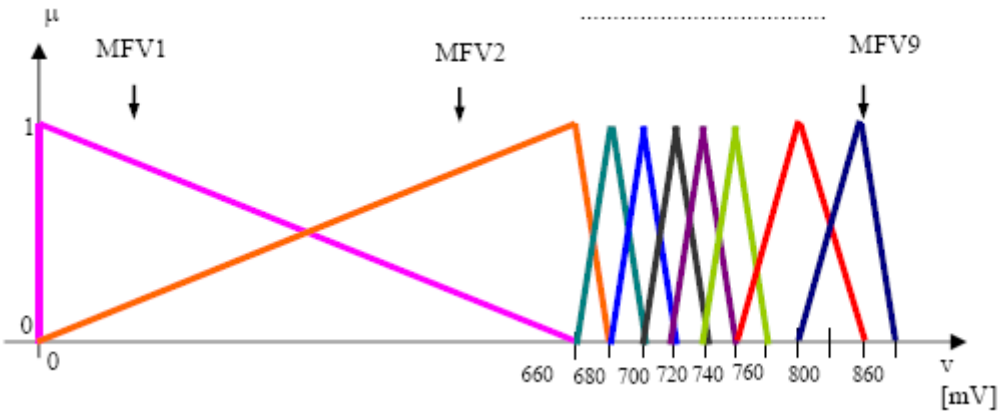


Figura 9. Mulţimile fuzzy peste universul discuţiei variabilei de intrare v

D2.2.2. Definirea mulţimilor fuzzy pentru variabila de ieşire i

Universul discuţiei variabilei de ieşire i, poate fi citit de pe caracteristica $i-v$ din Figura 6, şi este intervalul de numere reale: [0 ... 450 mA].
Acoperirea caracteristicii $i-v$ prin cele 9 "zone fuzzy" din Figura 6 presupune acoperirea universului discuţiei variabilei i prin 9 mulţimi fuzzy de formă triunghiulară; pentru simplitatea notaţiei, cele 9 mulţimi fuzzy sunt denumite: MFI1, MFI2, ..., MFI9. (MFI este o notaţie prescurtată pentru Mulţime Fuzzy pentru curentul I; indicele numeric corespunde "zonei fuzzy" respective).
Cele 9 mulţimi fuzzy nu mai formează o partiţie fuzzy; coordonatele suporturilor şi valorilor lor mediane pot fi citite din Figura 10.

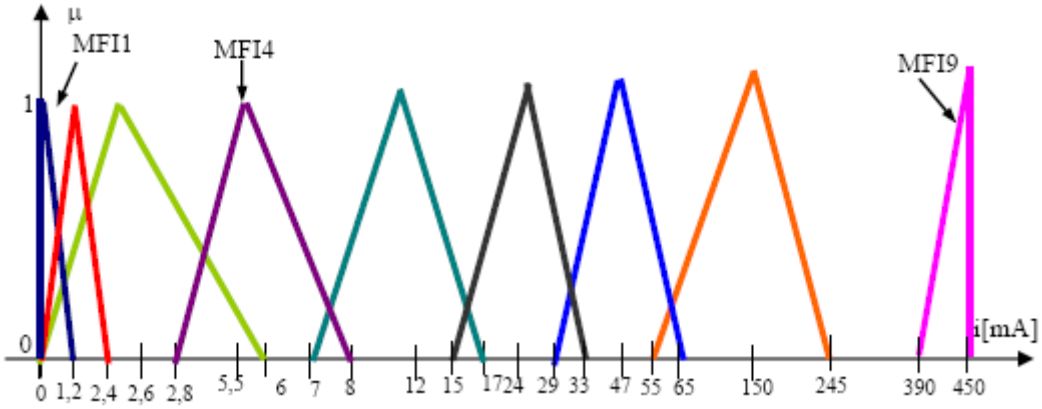


Figura 10. Mulţimile fuzzy peste universul discuţiei variabilei de ieşire i.

D2.2.3. Baza de reguli a sistemului cu logică fuzzy

Fiecare regulă din baza de reguli a sistemului cu logică fuzzy stabileşte o relaţie fuzzy între o mulţime fuzzy de intrare şi o mulţime fuzzy de ieşire, arătând valoarea fuzzy a ieşirii i, MFIk, corespunzătoare fiecărei valori fuzzy a intrării v, MFVj, k=1...9, j=1...9. Vom avea 9 reguli fuzzy în baza de reguli a sistemului; fiecare regulă fuzzy defineşte una din cele 9 "zone fuzzy" din Figura 1.
Cele 9 reguli fuzzy sunt:
R1: Dacă v este MFV1 atunci i este MFI1.
R2: Dacă v este MFV2 atunci i este MFI2.
.....
R9: Dacă v este MFV9 atunci i este MFI9.

D2. 2.4. Operaţia de fuzzificare

Fuzzificarea este de tip singleton, aşa cum a fost descrisă la punctul C2.

D2.2.5. Operaţia de inferenţă

Operaţia de inferenţă (cunoscută în literatura de specialitate sub numele de mecanism de inferenţă) folosită în această aplicaţie este de tip maxmin.

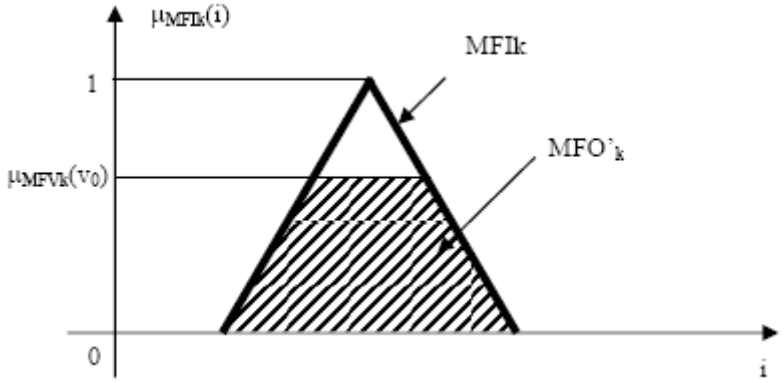


Figura 11. Obţinerea concluziei parţiale din regula fuzzy Rk.

Formula relaţiei de implicaţie (folosită pentru descrierea matematică a fiecărei reguli din baza de reguli fuzzy) este min (adică, Mamdani).
Cu aceste definiţii pentru mecanismul de inferenţă fuzzy şi implicaţia fuzzy, formula după care se evaluează mulţimea fuzzy de ieşire a oricărei reguli fuzzy: Rk: Dacă v este MFVk atunci i este MFIk (k=1...9) este:

$$\mu_{MFO'k}(i) = \max_{v} \min\left(\mu_{MFSI_0}(v), \min\left(\mu_{MFVk}(v), \mu_{MFIk}(i)\right)\right)$$

ceea ce, în cazul în care valoarea tranşantă de intrare se notează prin $v = v_0$, se reduce la:

$$\mu_{MFO'k}(i) = \min\left(\mu_{MFSI_0}(v_0), \mu_{MFIk}(i)\right)$$

unde prin $MFO'k$ se notează mulţimea fuzzy de ieşire generată de regula Rk, aşa cum se vede în Figura 11, sau, concluzia parţială a sistemului cu logică fuzzy.
Metoda de agregare a concluziilor parţiale $MFO'k$, k=1...9, date de cele 9 reguli din baza de reguli fuzzy, este max:

$$\mu_{MFO'}(i) = \max_{k} \mu_{MFO'k}(i) \quad \left(MFO' = \bigcup_{k} MFO'k\right).$$

D2.2.6. Operaţia de defuzzificare

Operaţia de selecţie a valorii tranşante i_0 a ieşirii, adică, de defuzzificare a rezultatului fuzzy MFO' al inferenţei fuzzy, este centrul maximelor, COM (=Center Of Maxima). Această operaţie înseamnă media ponderată a maximelor fiecărei concluzii parţiale $MFO'k$, unde ponderile sunt valorile maxime ale gradelor de apartenenţă din fiecare $MFO'k$.

D3. Desfăşurarea lucrării:

1. Bazându-vă pe cunoştinţele teoretice privind dezvoltarea unui sistem cu logică fuzzy în Matlab dobândite în cadrul punctului (C) al acestei lucrări, implementaţi acest sistem cu logică fuzzy în Matlab şi salvaţi-l sub numele: Dioda.fis.

2. Capturaţi fiecare din cele trei ferestre de editare grafică a sistemului cu logică fuzzy din Matlab, sub forma a trei grupuri de fişiere bitmap, care trebuie să conţină:
a) diagrama-bloc a sistemului cu logică fuzzy şi descrierea generală a sistemului – salvată în fişierul SLFGeneral.bmp;
b) forma şi parametrii funcţiilor de apartenenţă pentru fiecare variabilă (intrare şi ieşire) a sistemului cu logică fuzzy – salvate în fişierele FctApartenentaIn.bmp şi respectiv FctApartenentaOut.bmp;
c) regulile fuzzy ale sistemului, în format verbal – salvate în fişierul RuleBase.bmp.
3. Simulaţi sistemul cu logică fuzzy proiectat prin selectarea, din meniul View al oricăruia din cele trei editoare grafice, a comenzii Surface View. Salvaţi graficul $i(v)$ obţinut într-un fişier bitmap după ce în prealabil aţi capturat imaginea întregii ferestre Surface View; fişierul va avea numele CarIVDioda.bmp;
4. Verificaţi funcţionarea mecanismului de inferenţă în sistemul cu logică fuzzy proiectat prin selectarea, din meniul View al oricăruia din cele trei editoare grafice, a comenzii Rule View. Pentru diferite valori ale mărimii tranşante de intrare v, analizaţi: concluziile parţiale obţinute prin activarea fiecărei reguli din baza de reguli fuzzy; rezultatul agregării concluziilor parţiale; valoarea mărimii tranşante de ieşire i. Toate aceste rezultate analizate vor fi fie desenate şi respectiv notate pe hârtie, fie salvate într-un fişier bitmap prin acelaşi procedeu de captură a ferestrei.
5. Modificaţi operatorul de implicaţie la produs, din editorul sistemului cu logică fuzzy (fereastra FIS Editor); salvaţi sistemul cu logică fuzzy ca o nouă variabilă în spaţiul de lucru, cu numele Dioda1.
6. Repetaţi punctele 4 şi 5 de mai sus pentru noul sistem cu logică fuzzy, Dioda1, creat. Toate fişierele salvate vor avea în plus cifra 1 înainte de extensie.

E. Întrebări

Referatul lucrării de laborator va conţine răspunsurile voastre la următoarele întrebări:
1. Pe baza experimentelor derulate la punctul (D), realizaţi cate o scurtă descriere (maxim o pagină) a fiecăreia din cele două ferestre de depanare/verificare a unui sistem cu logică fuzzy puse la dispoziţie de componenta fuzzy a mediului Matlab.
2. Pe baza rezultatelor experimentale notate la punctele D2 - 4 şi D2 – 6 şi a caracteristicii experimentale a diodei, estimaţi precizia de aproximare a sistemului cu logică fuzzy proiectat, pentru fiecare din cele două implicaţii folosite.
3. Descrieţi matematic formula inferenţei folosite în fiecare din cele două cazuri de experimentare, D2 –4;-5 şi D2 –6;-7, conform teoriei mecanismelor de inferenţă studiate. Cărui tip de inferenţă studiat îi corespunde fiecare din cele două cazuri?
Încercaţi să găsiţi o altă explicare a mecanismului de inferenţă care să corespundă terminologiei folosite în Matlab, adică: inferenţă Mamdani pentru ambele situaţii, şi implicaţia: min, într-un caz, şi produs, în celălalt caz.

Lucrarea 2
Defuzificarea într-un sistem cu logică fuzzy.
Aplicaţie: maşina de spălat cu reguli fuzzy

A. Obiective

1) "Vizualizarea" procesului de selecţie a valorii tranşante de ieşire din mulţimea fuzzy de ieşire generată prin inferenţa fuzzy.
2) Observarea efectului folosirii unor metode de defuzzificare diferite asupra valorii tranşante obţinute ca ieşire, la aceeaşi mulţime fuzzy de ieşire.
3) Înţelegerea defuzificării ca şi operaţie de selecţie a unei singure valori tranşante reprezentative dintr-o mulţime fuzzy.

B. Concepte teoretice ilustrate

Lucrarea de laborator se bazează pe ***conceptele teoretice de bază privind structura şi operaţiile unui sistem cu logică fuzzy***, accentul punându-se în această lucrare pe ***înţelegerea amănunţită a operaţiei matematice de defuzificare*** şi pe cunoaşterea unor ***formule matematice de defuzificare*** frecvent folosite în practică. Toate aceste concepte au fost studiate la Cursul de Sisteme de Reglare Fuzzy.
Pentru aprofundarea acestor concepte, vi se recomandă citirea:
1) von Altrock, C., *Fuzzy Logic and NeuroFuzzy Applications Explained*, Prentice Hall, Englewood Cliffs, New Jersey, 1995
2) Fuller, R., *Neural Fuzzy Systems*, Abo Akademi University, Finlanda, 1995

C. Prezentarea aplicaţiei practice

Una dintre aplicaţiile cele mai importante în practică ale sistemelor cu logică fuzzy se referă la folosirea acestora ca ***sisteme de reglare a proceselor***. Folosirea sistemelor cu logică fuzzy pentru sinteza regulatoarelor se bucură de o bază teoretică solidă, existând la ora actuală multe aplicaţii comerciale care încorporează un sistem de reglare fuzzy. Un exemplu reprezentativ în acest sens este şi aplicaţia prezentată în lucrare, aleasă pentru a ilustra experimental ***operaţia de defuzificare*** – ultimul pas în obţinerea soluţiei tranşante generate de un sistem cu logică fuzzy.
Atunci când folosim o maşină de spălat, selectăm de obicei durata de spălare în funcţie de cantitatea de haine pe care dorim să le spălăm şi de tipul şi gradul de murdărie al acestor haine.
Pentru a automatiza procesul de selecţie a duratei de spălare, putem folosi senzori care să detecteze parametrii pe baza cărora omul însuşi selectează timpul necesar spălării (adică, senzori de detecţie a volumului hainelor de spălat, a gradului şi tipului de murdărie). Apoi, pe baza acestor date, se poate determina timpul de spălare. Din nefericire, este dificil sau aproape imposibil de formulat o relaţie matematică exactă între volumul hainelor şi gradul şi tipul de murdărie, pe de o parte, şi timpul de spălare necesar, pe de altă parte. Ca urmare, această problemă a rămas nerezolvată până de curând: utilizatorul trebuia pur şi simplu să seteze manual timpul de spălare, prin încercări şi pe baza experienţei. Astfel, maşinile de spălat nu atinseseră gradul lor maxim de automatizare. Această problemă o rezolvă folosirea logicii fuzzy, aşa cum se va vedea în continuare.
Pentru a construi o maşină de spălat complet automată cu timpi de spălare autodeterminabili, trebuie să ne concentrăm asupra construirii a două subsisteme esenţiale ale maşinii de spălat:
(1) sistemul senzorilor;

(2) unitatea de comandă.
Sistemul senzorilor furnizează semnalele de intrare ale maşinii de spălat, preluate din mediul exterior (hainele de curăţat); pe baza acestor semnale de intrare, maşina va lua decizia asupra duratei de spălare.
Unitatea de comandă are sarcina de a lua decizia asupra timpului necesar de spălare şi de a semnala această decizie în exterior sub forma unei ieşiri de comandă. Din cauza faptului că între intrarea şi ieşirea unităţii de comandă nu se poate formula o relaţie matematică clară, proiectarea ei nu se poate realiza prin metodele tradiţionale de reglare, dar în schimb se poate formula relativ uşor folosind logica fuzzy. De aceea, soluţia propusă în lucrare foloseşte ca şi unitate de comandă un ***sistem de reglare fuzzy***.

C1. Sistemul cu logică fuzzy pentru controlul timpului de spălare

Obiectivul aplicaţiei din lucrare este proiectarea unui sistem de reglare fuzzy pentru o maşină de spălat, care să dea ca ieşire timpul de spălare corect în funcţie de datele de intrare despre hainele care trebuie spălate, în condiţiile în care nu avem la dispoziţie un model precis al legăturii dintre intrarea şi ieşirea sistemului de reglare. Schema-bloc a regulatorului fuzzy este cea din Figura 1.

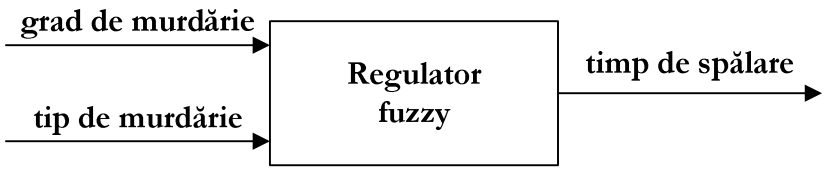


Figura 1. Schema-bloc a regulatorului fuzzy.

Sistemul de reglare fuzzy are două intrări: o intrare pentru gradul de murdărie a hainelor, şi a doua intrare pentru tipul de murdărie a hainelor. Valorile numerice ale celor două intrări pot fi obţinute de la un singur senzor optic:
(1) **gradul de murdărie** poate fi determinat din transparenţa apei în care se spală hainele; cu cât sunt mai murdare hainele, cu atât transparenţa apei va fi mai mică.
(2) **tipul de murdărie** poate fi determinat din timpul necesar apei în care se înmoaie hainele să ajungă cu transparenţa la saturaţie. Saturaţia transparenţei apei este momentul de timp după care modificarea transparenţei apei este aproape zero. De exemplu, hainele cu pete de grăsime necesită un timp mai lung până la saturarea transparenţei apei, deoarece grăsimea este mai greu solubilă decât alte forme de murdărie.
În această lucrare de laborator, vom analiza operaţia de ***defuzificare***, care selectează din rezultatul fuzzy (mulţime fuzzy) generat de operaţia de inferenţă o singură valoare tranşantă, reprezentativă pentru mulţimea fuzzy rezultată.
Modul de selecţie a acestei valori unice nu este o problemă simplă; există mai multe variante posibile de selecţie, şi succesul uneia sau alteia dintre ele depinde de aplicaţia concretă pe care dorim s-o rezolvăm. Totuşi, din fericire, în general toate aplicaţiile din aceeaşi clasă sunt rezolvate bine de una sau două metode de defuzzificare standard, ceea ce uşurează procesul selecţiei metodei folosite.
Pentru a putea studia însă ***defuzificarea*** care are loc ca şi operaţia ultimă din lanţul de prelucrare al unui sistem cu logică fuzzy, este desigur nevoie să definim mai întâi componentele sistemului: mulţimi fuzzy de intrare, mulţimi fuzzy de ieşire, bază de reguli fuzzy, şi deasemenea să specificăm modul de definire a operaţiilor sistemului cu logică fuzzy: metoda de fuzzificare, mecanismul de inferenţă şi, abia în final, metoda de defuzzificare.

C1.1. Definirea mulţimilor fuzzy pentru variabila de intrare *grad de murdărie*

Universul discuţiei variabilei de intrare grad de murdărie, notată *gradmurdarie*, se defineşte în procente faţă de termenul subiectiv "murdar", şi este: 0...100 [%] murdar.
Pentru definirea mulţimilor fuzzy pentru variabila de intrare *gradmurdarie* se foloseşte cea mai simplă acoperire a universului discuţiei variabilei de intrare, prin 3 mulţimi fuzzy triunghiulare, două asimetrice (pe extremele universului discuţiei: **mic** şi **mare**) şi una simetrică (**mediu**), astfel încât cele 3 mulţimi să formeze o partiţie fuzzy. Valorile mediane ale celor 3 mulţimi fuzzy se aleg astfel: 0% pentru **mic**, 50% pentru **mediu** şi 100% pentru **mare**. Acest gen de alegere a mulţimilor fuzzy este uzual în cazul sistemelor cu logică fuzzy destinate reglării. Practica demonstrează că, deşi ar fi mai simplă acoperirea prin doar două mulţimi fuzzy a universului discuţiei variabilei de intrare, acoperirea cu trei mulţimi fuzzy conduce la rezultate mult mai bune, cu o complicaţie minoră a calculelor.
Cele 3 mulţimi fuzzy pentru variabila de intrare *gradmurdarie* sunt reprezentate în Figura 2.

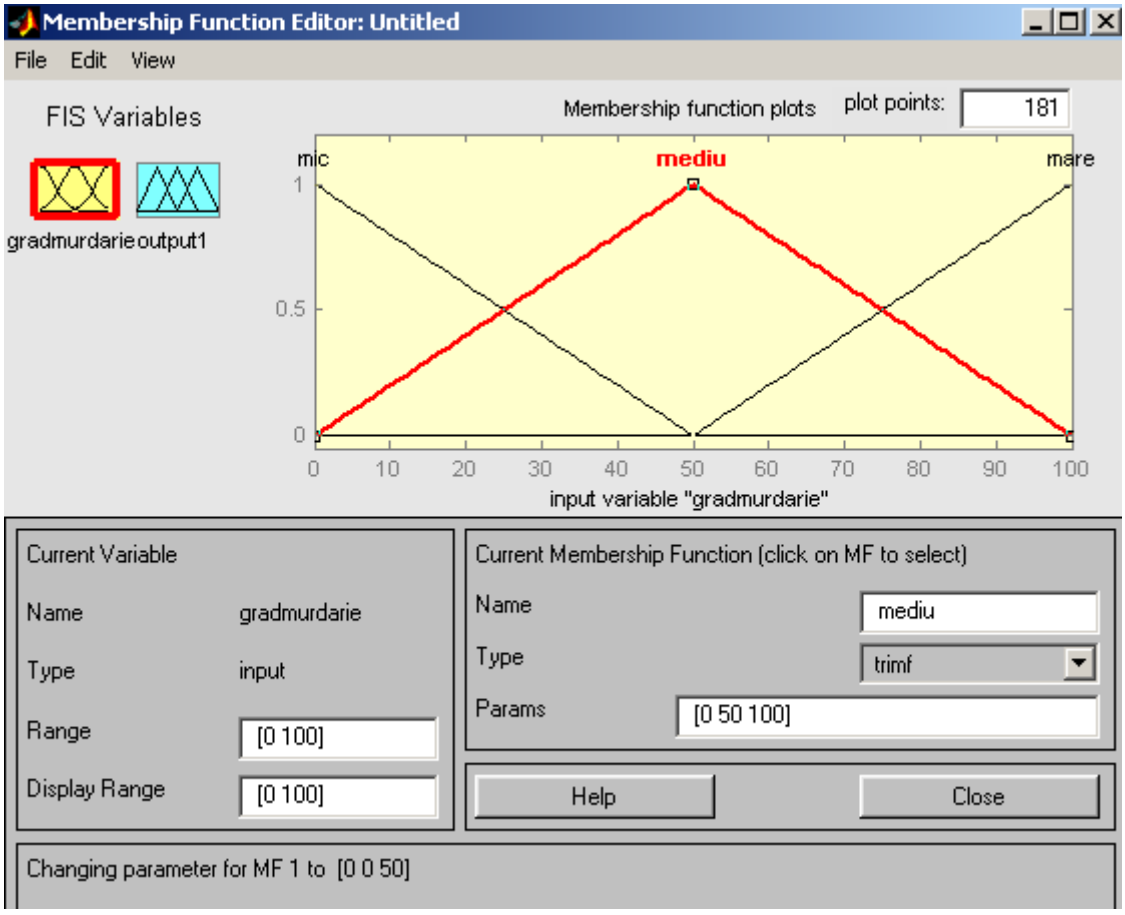


Figura 2. Mulţimile fuzzy peste universul discuţiei variabilei de intrare *gradmurdarie*.

C1.2. Definirea mulţimilor fuzzy pentru variabila de intrare *tip de murdărie*

• **Universul discuţiei** variabilei de intrare tip de murdărie, notată *tipmurdarie*, se defineşte în procente faţă de termenul "gras", şi este: 0-100 [%] gras.
• Din aceleaşi motive ca şi în cazul variabilei *gradmurdarie*, şi pentru definirea mulţimilor fuzzy pentru variabila de intrare *tipmurdarie* se foloseşte acoperirea universului discuţiei variabilei de intrare prin 3 mulţimi fuzzy triunghiulare, două asimetrice (pe extremele universului discuţiei: **negras** şi **gras**) şi una simetrică (**mediu**), astfel încât cele 3 mulţimi să formeze o partiţie

fuzzy. Valorile mediane ale celor 3 mulţimi fuzzy se aleg astfel: 0% pentru **negras**, 50% pentru **mediu** şi 100% pentru **gras**.
Cele 3 mulţimi fuzzy pentru variabila de intrare *tipmurdarie* sunt reprezentate în Figura 3.

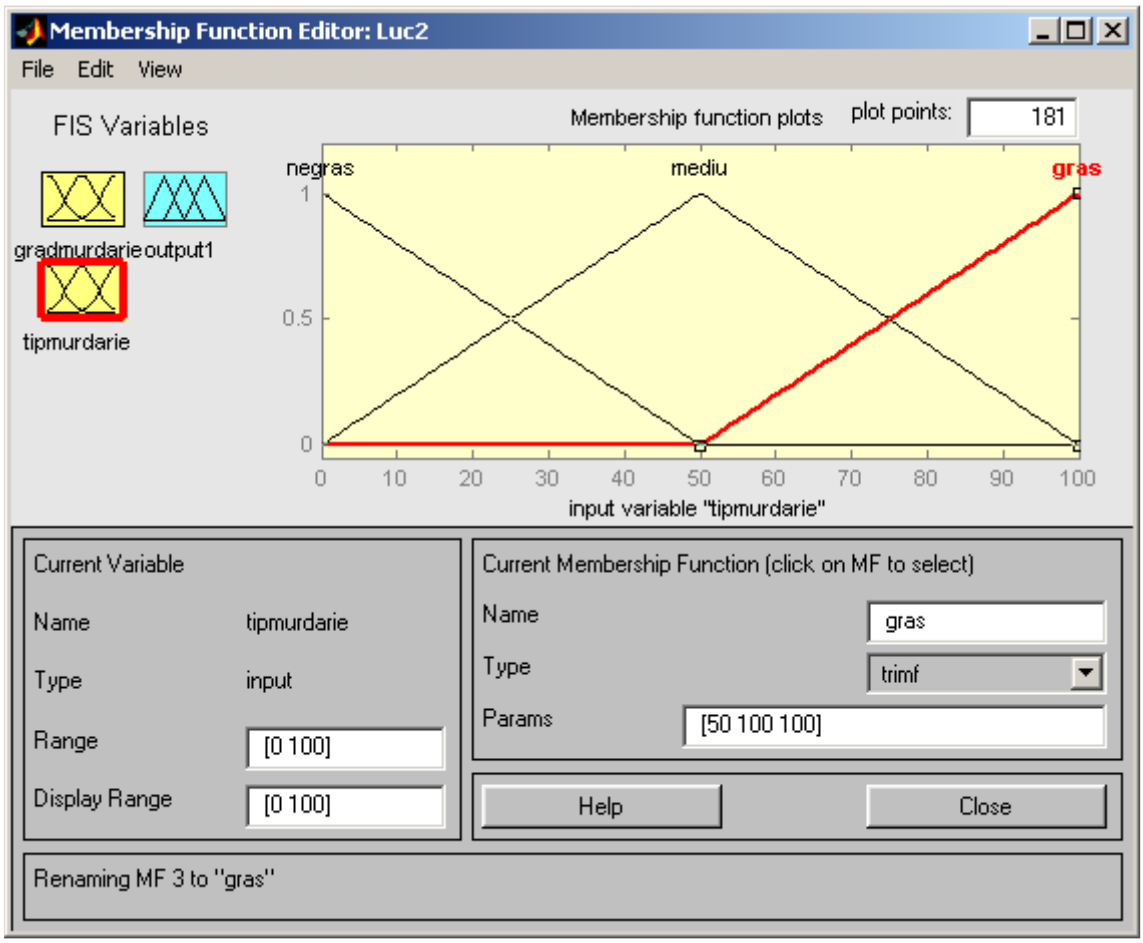


Figura 3. Mulţimile fuzzy peste universul discuţiei variabilei de intrare *tipmurdarie*

C1.3. Definirea mulţimilor fuzzy pentru variabila de ieşire *timp de spălare*

• **Universul discuţiei** variabilei de ieşire timp de spălare, notată *t,* se defineşte plecând de la observaţia practică a timpului maxim de spălare necesitat de o haină cu murdărie maximă, care nu depăşeşte o oră; ca urmare, universul discuţiei pentru t este: 0-60 [min].
• Pentru definirea mulţimilor fuzzy ale timpului de spălare *t*, se foloseşte tot observarea selecţiei practice a timpului de spălare de către utilizator, care ar putea clasifica duratele stabilite la spălarea hainelor în cinci categorii, în funcţie de gradul şi tipul de murdărie al hainei: timp foarte scurt de spălare (în jur de 8 minute); timp scurt de spălare (în jur de 12 minute); timp mediu de spălare (în jur de 20 de minute); timp lung de spălare (în jur de 40 de minute); timp foarte lung de spălare (în jur de 60 de minute).
În conformitate cu aceste aprecieri bazate pe experienţa practică a operatorului uman, descrierea valorilor posibile pentru variabila de ieşire *t* se face în sistemul cu logică fuzzy de reglare proiectat în lucrare prin 5 mulţimi fuzzy triunghiulare, definite astfel încât să formeze o partiţie fuzzy (excepţie făcând capetele universului discuţiei). Cele cinci mulţimi fuzzy pentru *t* sunt numite: **foartescurt, scurt, mediu, lung** şi **foartelung**, şi sunt reprezentate în Figura 4.

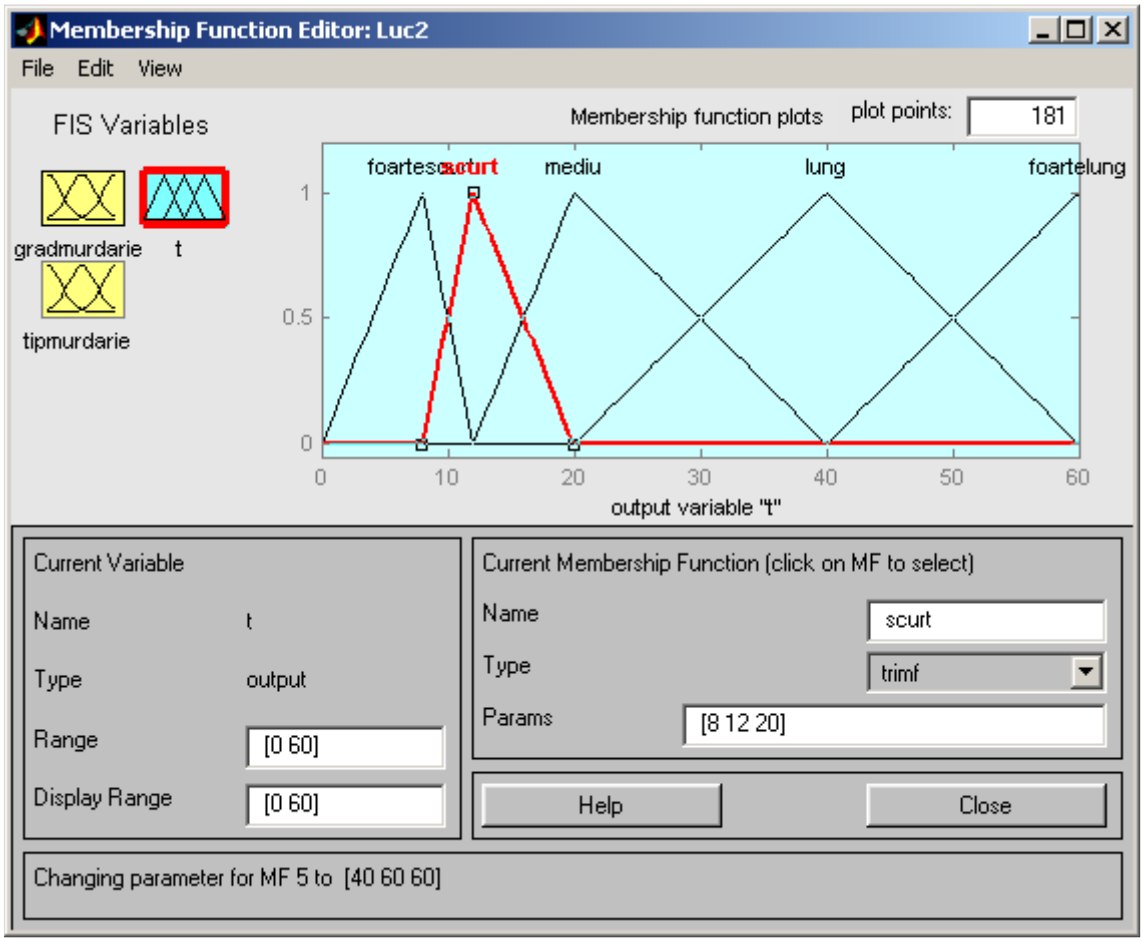


Figura 4. Mulţimile fuzzy peste universul discuţiei variabilei de ieşire, *t*

Observaţie: valorile mediane şi suporturile celor cinci mulţimi fuzzy sunt, în ordine:
foartescurt: [0 8 12]**; scurt:** [8 12 20]; **mediu** [12 20 40]; **lung:** [20 40 60];
foartelung: [40 60 60]

C1.4. Baza de reguli a sistemului de reglare fuzzy

Regulile de tip "*Dacă....atunci....*" ale sistemului de reglare fuzzy reflectă capacitatea de decizie a sistemului. Regulile sunt afirmaţii calitative deduse de către proiectantul sistemului de reglare fuzzy din datele şi "experimentele" realizate in procesul de spălare a hainelor cu maşina de spălat.

Pe această bază, o enunţare intuitivă corectă a regulilor fuzzy pentru aplicaţia din lucrare este următoarea:

Dacă *gradmurdarie* este mare ŞI *tipmurdarie* este gras ***Atunci*** *t* este foartelung.
Dacă *gradmurdarie* este mediu ŞI *tipmurdarie* este gras ***Atunci*** *t* este lung.
Dacă *gradmurdarie* este mic ŞI *tipmurdarie* este gras ***Atunci*** *t* este lung.
Dacă *gradmurdarie* este mare ŞI *tipmurdarie* este mediu ***Atunci*** *t* este lung.
Dacă *gradmurdarie* este mediu ŞI *tipmurdarie* este mediu ***Atunci*** *t* este mediu.
Dacă *gradmurdarie* este mic ŞI *tipmurdarie* este mediu ***Atunci*** *t* este mediu.
Dacă *gradmurdarie* este mare ŞI *tipmurdarie* este negras ***Atunci*** *t* este mediu**.**
Dacă *gradmurdarie* este mediu ŞI *tipmurdarie* este negras ***Atunci*** *t* este scurt.
Dacă *gradmurdarie* este mic ŞI *tipmurdarie* este negras ***Atunci*** *t* este foartescurt.

C1.5. Operaţia de fuzzificare

Fuzificarea este de tip singleton, adică valoarea tranşanta de intrare este transformată într-o mulţime fuzzy a cărei suport este un singur punct din universul discuţiei (valoarea tranşantă), cu grad de apartenenţă 1.

C1.6. Operaţia de inferenţă

Operaţia de inferenţă (cunoscută în literatura de specialitate sub numele de mecanism de inferenţă) în această aplicaţie va fi de tip ***max-min.*** **Formula relaţiei de implicaţie** (folosită pentru descrierea matematică a fiecărei reguli din baza de reguli fuzzy) poate fi ***min*** (adică, Mamdani), sau ***prod*** (produs, adica Larsen). **Metoda de agregare** a concluziilor parţiale date de regulile din baza de reguli fuzzy este ***max***..

C1.7. Operaţia de defuzzificare

Operaţia de selecţie a valorii tranşante a ieşirii, adică, de **defuzzificare** a rezultatului fuzzy al inferenţei fuzzy, este ***centrul ariei, COA*** (=Center Of Area). Această operaţie este foarte cunoscută în literatură sub numele de ***defuzzificare centroid***, şi este cea mai frecvent folosită metodă de defuzzificare pentru aplicaţiile de reglare, deoarece succesiunea răspunsurilor tranşante obţinute prin acest tip de defuzzificare este suficient de lină pentru a asigura o curbă de reglare fără variaţii bruşte. Ori, este cunoscut de la ingineria sistemelor automate că aceasta este o cerinţă importantă a unui regulator.

Formula după care se calculează valoarea tranşantă a ieşirii, pe baza mulţimii fuzzy de ieşire, în cazul folosirii metodei de defuzzificare COA (centroid) este următoarea (v. Figura 5):

$$t_0 = \frac{\int_{t=0}^{60} t \cdot \mu_{MFO}(t)}{\int_{t=0}^{60} \mu_{MFO}(t)}, \text{ iar pentru cazul discret } t_0 = \frac{\sum_{t=0}^{60} t \cdot \mu_{MFO}(t)}{\sum_{t=0}^{60} \mu_{MFO}(t)}.$$

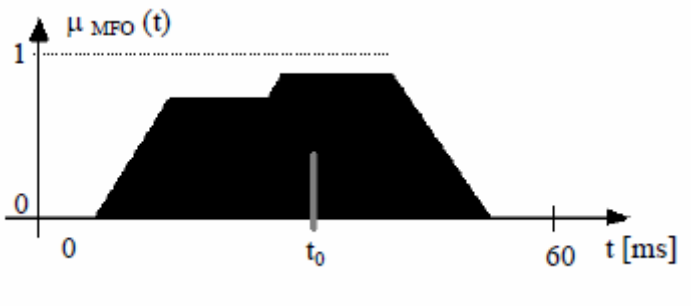


Figura 5. Rezultatul – mulţime fuzzy de ieşire – al operaţiei de inferenţă fuzzy, defuzificat prin metoda COA (centroid)

D. Partea experimentală

D1. Indicaţii preliminare:

• Pentru partea experimentală, veţi folosi mediul software care permite simularea sistemelor cu logică fuzzy FIS Editor din Matlab. Sistemul cu logică fuzzy va fi descris printr-un fişier *.fis.

• Experimentele presupun: construirea grafică a sistemului fuzzy de reglare a timpului de spălare în Matlab şi salvarea sa în fişierul MasSpalat.fis; simularea şi analiza sistemului cu logică

fuzzy în mediul Matlab, folosind instrumentele de vizualizare a suprafeţei de reglare a sistemului şi a regulilor fuzzy; salvarea rezultatelor experimentale în fişiere de tip bitmap şi notarea în tabele a valorilor măsurate prin simulare. La punctul **E** al lucrării, veţi examina rezultatele experimentale obţinute cu scopul de a răspunde la setul de întrebări formulate.

D2. Desfăşurarea lucrării:

1. Construiţi sistemul fuzzy MasSpalat.fis şi salvaţi-l pe hard-disk:
    - 1.1. Definiţi variabilele de intrare şi de ieşire
    - 1.2. Definiţi mulţimile fuzzy peste fiecare variabilă (domeniul de valori, numărul de mulţimi fuzzy, parametrii mulţimilor fuzzy)
    - 1.3. Editaţi regulile
    - 1.4. Salvaţi fişierul
2. Analizaţi funcţionarea SRF vizualizând suprafaţa de reglare (View/Surface) şi operaţiile realizate in sistemul de reglare fuzzy
    - 2.1. Comparaţi suprafeţele de reglare şi valorile obţinute la ieşire în două variante:
        - 2.1.1. Implication: ***min***; Defuzzification: ***centroid***
        - 2.1.2. Implication: ***prod***; Defuzzification: ***centroid***
    - 2.2. Comparaţi suprafeţele de reglare şi valorile obţinute la ieşire în variantele:
        - 2.2.1. Implication: ***min***; Defuzzification: ***centroid***
        - 2.2.2. Implication: ***min***; Defuzzification: ***bisector***
        - 2.2.3. Implication: ***min***; Defuzzification: ***MOM***
    - 2.3. Pentru aceasta creaţi un document word în care veţi plasa rezultatele experimentelor:
        - 2.3.1. Suprafeţele de reglare pentru toate cele 4 cazuri de mai sus
        - 2.3.2. Completaţi un tabel cu valorile obţinute la ieşire, pentru cel puţin 7 combinaţii semnificative ale mărimilor de intrare, pentru toate cele 4 cazuri de mai sus

	In1	In2	Implicaţie	Defuzificare	Ieşire
1			min	centroid	
			prod	centroid	
			min	bisector	
			min	Mom	
2			min	centroid	
			prod	centroid	
			min	bisector	
			min	Mom	
3			min	centroid	
			prod	centroid	
			min	bisector	
			min	Mom	
4			min	centroid	
			prod	centroid	
			min	bisector	
			min	Mom	
5			min	centroid	
			prod	centroid	
			min	bisector	
			min	Mom	

	In1	In2	Implicaţie	Defuzificare	Ieşire
6			min	centroid	
			prod	centroid	
			min	bisector	
			min	Mom	
7			min	centroid	
			prod	centroid	
			min	bisector	
			min	Mom	

E. Interpretarea rezultatelor experimentale

Referatul lucrării de laborator va conţine răspunsurile voastre la următoarele întrebări:
1. De ce suprafaţa de reglare a unui sistem de reglare în general trebuie să fie cât mai netedă? Examinaţi, din această perspectivă, caracteristicile *t(gradmurdarie, tipmurdarie)* obţinute cu fiecare metodă de defuzzificare.
Care metodă de defuzzificare este cea mai potrivită, şi care este cea mai defavorabilă pentru implementarea sistemului fuzzy de reglare din aplicaţia studiată? Justificaţi şi matematic răspunsul, examinând formula de calcul a ieşirii tranşante prin fiecare din metodele de defuzzificare studiate.
2. Pentru 2 din perechile de valori tranşante de intrare selectate la punctul **D2**, desenaţi operaţiile realizate în SRF, precum şi mulţimea fuzzy de ieşire rezultată în urma inferenţei fuzzy, în cazul defuzificării centroid.
3. Repetaţi punctul 2 pentru metoda de defuzzificare MOM.
4. Dacă ar fi să precizaţi, pe baza intuiţiei, o valoare tranşantă a timpului de spălare pentru fiecare pereche de valori tranşante de intrare de la punctul **D2**, care ar fi aceasta în fiecare caz şi cât de apropiată/diferită este ea de valorile ieşirii obţinute prin fiecare metodă de defuzzificare studiată?
5. Există o legătură de principiu între selecţia valorii tranşante de ieşire prin defuzzificare şi alegerea centrelor mulţimilor fuzzy prin algoritmul fuzzy cmeans? Care este aceasta?

Lucrarea 3
Aplicaţia 1: regulator fuzzy pentru fluidizarea traficului
Aplicaţia 2: regulator fuzzy pentru nivelul glicemiei

A. Obiective

1) "Vizualizarea" procesului de selecţie a valorii tranşante de ieşire din mulţimea fuzzy de ieşire generată prin inferenţa fuzzy.
2) Observarea efectului folosirii unor metode de defuzzificare diferite asupra valorii tranşante obţinute ca ieşire, la aceeaşi mulţime fuzzy de ieşire.
3) Aprecierea modului de constituire a unei baze de reguli..

B. Concepte teoretice ilustrate

Lucrarea de laborator se bazează pe ***conceptele teoretice de bază privind structura şi operaţiile unui sistem cu logică fuzzy***, accentul punându-se în această lucrare pe ***înţelegerea amănunţită a operaţiei matematice de defuzificare*** şi pe cunoaşterea unor ***formule matematice de defuzificare*** frecvent folosite în practică.

C. Prezentarea aplicaţiei practice 1

În condiţiile în care numărul maşinilor este în continuă creştere, iar infrastructură nu este în măsură să facă faţă fluxului uriaş de autovehicule, în tot mai multe intersecţii apar blocaje.
Pentru fluidizarea traficului, trebuie să avem un sistem de conducere al circulaţiei care să poată să optimizeze timpul de aşteptare la fiecare intersecţie, în mod automat, în funcţie de numărul de maşini aflate deja în intersecţie şi de numărul celor ce vor sosi.

C.1 Mărimile de intrare şi de ieşire

Mărimile de intrare sunt :

- Numărul de maşini aflate deja în intersecţie QUEUE;
- Numărul de maşini ce vor sosi ARRIVAL.

Mărimea de ieşire este :

- Intervalul de timp al semnalului roşu la o intersecţie EXTENSION.

C.1.1 Domeniile de valori ale mărimilor de intrare şi ieşire

ARRIVAL [0 6]
QUEUE [0 6]
EXTENSION [0 12]

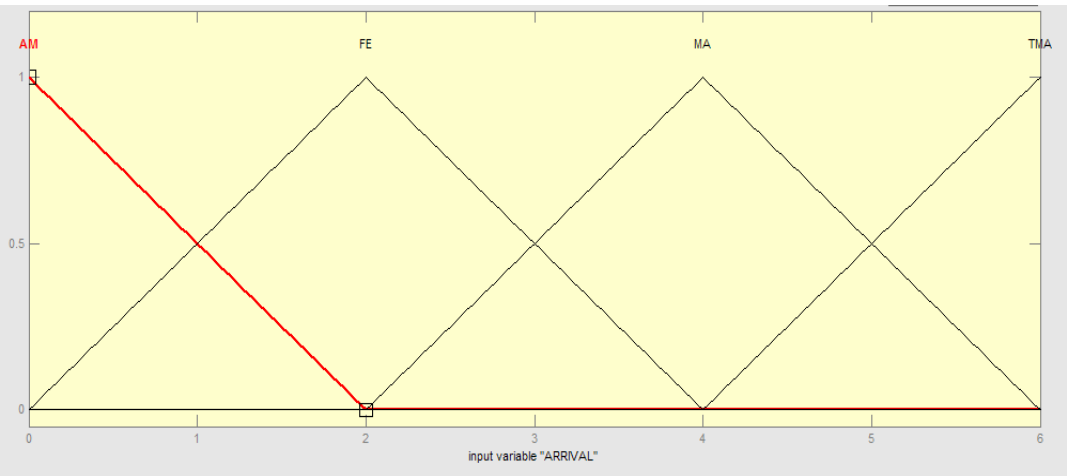
C.1.2 Variabile lingvistice şi funcţii de apartenenţă

Pentru variabilele lingvistice ale regulatorului se aleg următorii termeni lingvistici :

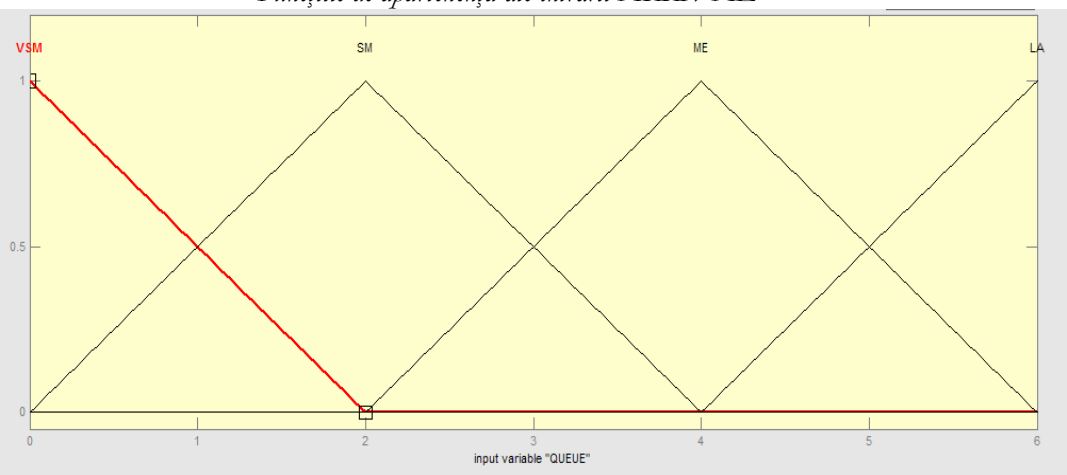
- Pentru ARRIVAL : ALMOST (AL), FEW (FE), MANY (MA), TOO MANY (TMA)

- Pentru QUEUE : VERY SMALL (VSM), SMALL (SM), MEDIUM (ME), LARGE (LA)
- Pentru EXTENSION : ZERO (ZE), SHORT (SH), MEDIUM (ME), LONGER (LO)

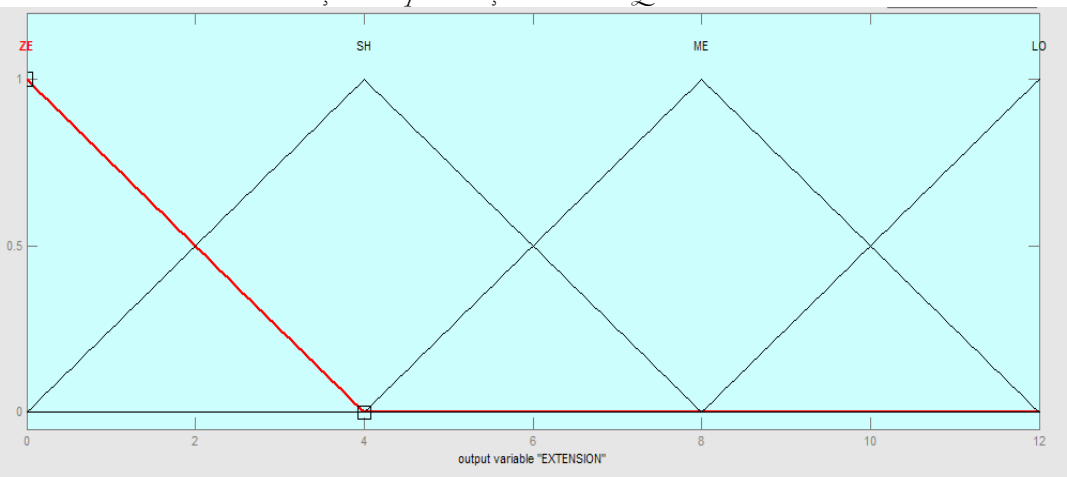
Funcţiile de apartenenţă se aleg conform figurilor :



Funcţiile de apartenenţă ale intrării ARRIVAL



Funcţiile de apartenenţă ale intrării QUEUE



Funcţiile de apartenenţă ale ieşirii EXTENSION

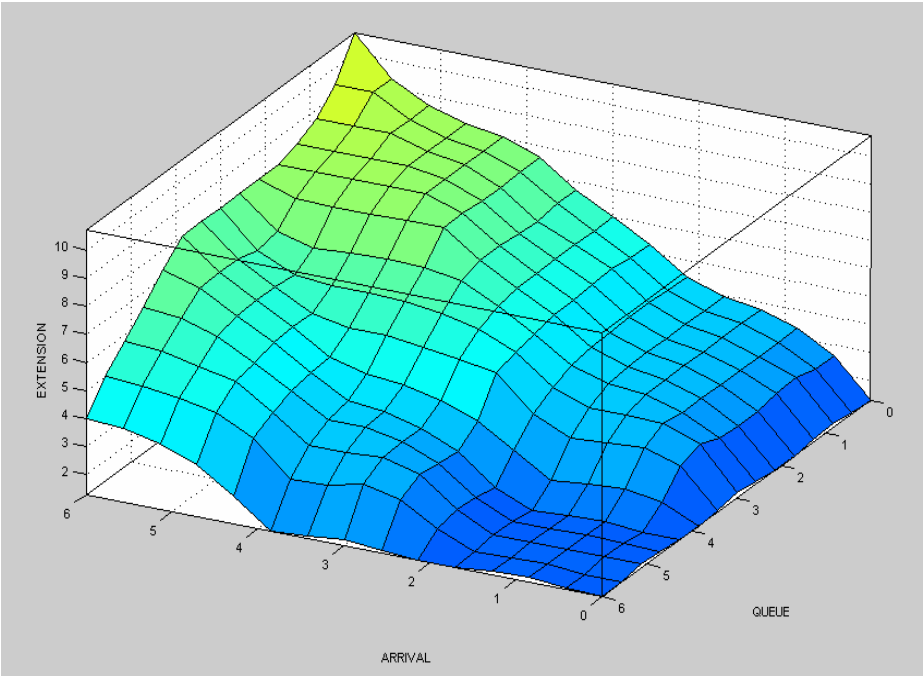
C.1.3 Baza de reguli

a. Dacă ARRIVAL este AL şi QUEUE este VSM, atunci EXTENSION este ZE,
b. Dacă ARRIVAL este AL şi QUEUE este SM, atunci EXTENSION este ZE,
c. Dacă ARRIVAL este AL şi QUEUE este ME, atunci EXTENSION este ZE,
d. Dacă ARRIVAL este AL şi QUEUE este LA, atunci EXTENSION este ZE,
e. Dacă ARRIVAL este FE şi QUEUE este VSM, atunci EXTENSION este SH,
f. Dacă ARRIVAL este FE şi QUEUE este SM, atunci EXTENSION este SH,
g. Dacă ARRIVAL este FE şi QUEUE este ME, atunci EXTENSION este ZE,
h. Dacă ARRIVAL este FE şi QUEUE este LA, atunci EXTENSION este ZE,
i. Dacă ARRIVAL este MA şi QUEUE este VSM, atunci EXTENSION este ME,
j. Dacă ARRIVAL este MA şi QUEUE este SM, atunci EXTENSION este ME,
k. Dacă ARRIVAL este MA şi QUEUE este ME, atunci EXTENSION este SH,
l. Dacă ARRIVAL este MA şi QUEUE este LA, atunci EXTENSION este ZE,
m. Dacă ARRIVAL este TMA şi QUEUE este VSM, atunci EXTENSION este LO,
n. Dacă ARRIVAL este TMA şi QUEUE este SM, atunci EXTENSION este ME,
o. Dacă ARRIVAL este TMA şi QUEUE este ME, atunci EXTENSION este ME,
p. Dacă ARRIVAL este TMA şi QUEUE este LA, atunci EXTENSION este SH.

		QUEUE			
		VSM	SM	ME	LA
ARRIVAL	AL	ZE	ZE	ZE	ZE
	FEW	SH	SH	ZE	ZE
	MA	ME	ME	SH	ZE
	TMA	LO	ME	ME	SH

C.2 Suprafaţa de reglare

Ea prezintă dependenţa mărimii de ieşire de mărimile de intrare. Astfel, suprafaţa de reglare reflectă modul de comportare a regulatorului pentru orice pereche de valori ferme de intrare.



Prezentarea de mai sus ne arată un exemplu de reglare fuzzy pentru o singură intersecţie. În realitate, când construim un sistem de fluidizare a traficului, trebuie să sincronizăm toate intersecţiile consecutive astfel încât traficul să fie cât mai convenabil.

D. Prezentarea aplicaţiei practice 2

Diabetul este una dintre cele mai răspândite boli din zilele noastre. Această boală este caracterizată de secreţia insuficientă de insulină de către pancreas datorită unei disfuncţii a celulelor β. Astfel, se poate ajunge la o creştere a concentraţiei de glucoză din sânge peste limita de 4.5 mmol/L. Pentru a putea îmbunătăţi terapia diabetului este necesară o cunoaştere amănunţită a dinamicii interacţiunii dintre insulină şi glucoză. În ultimii 30 de ani a fost dezvoltat un model matematic neliniar ce exprimă interacţiunea dintre insulină şi glucoză în corpul unui pacient diabetic. Modelul constă într-o serie de ecuaţii diferenţiale neliniare care sunt rezolvate în general, numeric. Metoda convenţională de a evalua acest model ca pe unul rigid, clasic se dovedeşte a nu fi atât de bună pentru că sistemele biologice sunt supuse incertitudinii. În primul rând parametrii modelului sunt incerţi; ei pot varia atât în funcţie de constituţia fizică a pacientului, cât şi în funcţie de durata bolii. În al doilea rând, intrarea modelului, care constă în valoarea nutriţională a alimentelor consumate de pacient, poate fi estimată cu un grad destul de mare de imprecizie. În general, sistemele biologice sunt foarte complexe şi supuse incertitudinii, deci metodele convenţionale de reglare se pot dovedi neeficiente.
O abordare care poate rezolva aceste probleme o reprezintă folosirea regulatoarelor fuzzy pentru a ţine cont de incertitudinile ce pot apărea în dinamica glucoză-insulină. În esenţă, parametrii şi intrările modelului sunt reprezentate de numere fuzzy, cu forma aleasă în urma datelor experimentale sau a cunoştinţelor experţilor.

Principalul scop al regulatorului fuzzy este acela de a menţine un nivel normal al glucozei în sânge(aprox 4.5 mmol/L).

D.1 Mărimile de intrare şi de ieşire

Mărimile de *intrare* sunt:

- Nivelul glicemiei din sânge G(t) definită ca:

- Viteza de variaţie a glicemiei, $\dot{G}(t)$

Mărimea de *ieşire* este reprezentată de doza de insulină I, injectată subcutanat.

D.1.1. Domeniile de valori ale mărimilor de intrare şi ieşire

Pentru mărimile de intrare se aleg următoarele domenii de variaţie:

$G(t) \in [0, 16]$ mmol/L

$\dot{G}(t) \in [-25, 25]$ mmol/L*h

Pentru mărimea de ieşire se alege următorul domeniu de variaţie:

$I \in [400, 3000]$ mU/h

D.1.2. Variabile lingvistice şi funcţii de apartenenţă

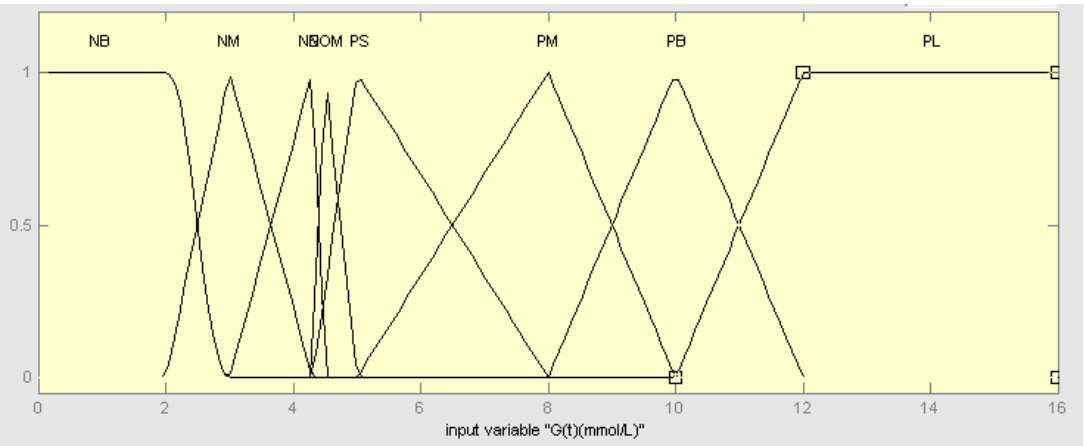
Pentru variabilele lingvistice ale regulatorului se aleg următorii termeni lingvistici:

● pentru G(t): NB(negative big) , NM(negative meduim), NS(negative small), NOM(normal), PS(positive small), PM(positive medium), PB(positive big), PL(positive large);

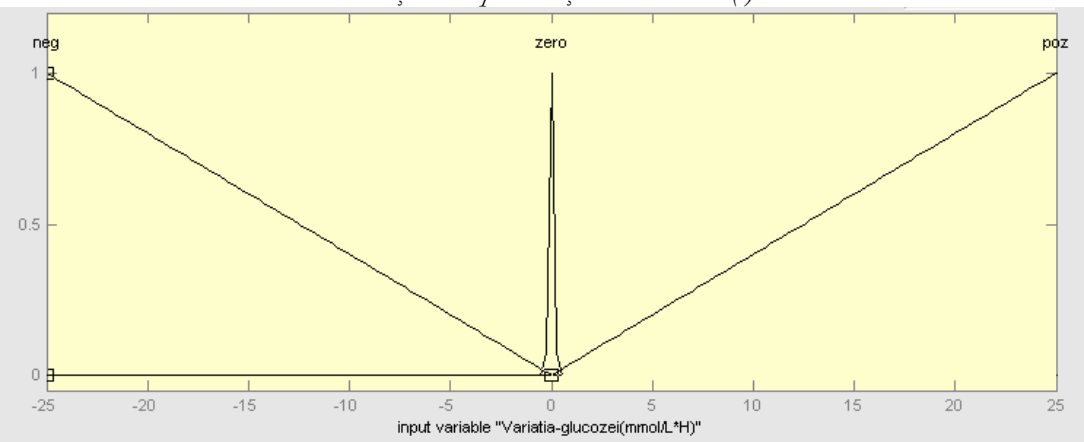
● pentru $\dot{G}(t)$: neg(negative), zero, pos(positive);

● pentru I: NB(negative big) , NM(negative meduim), NS(negative small), Z(zero), PS(positive small), PM(positive medium), PB(positive big), PL(positive large);

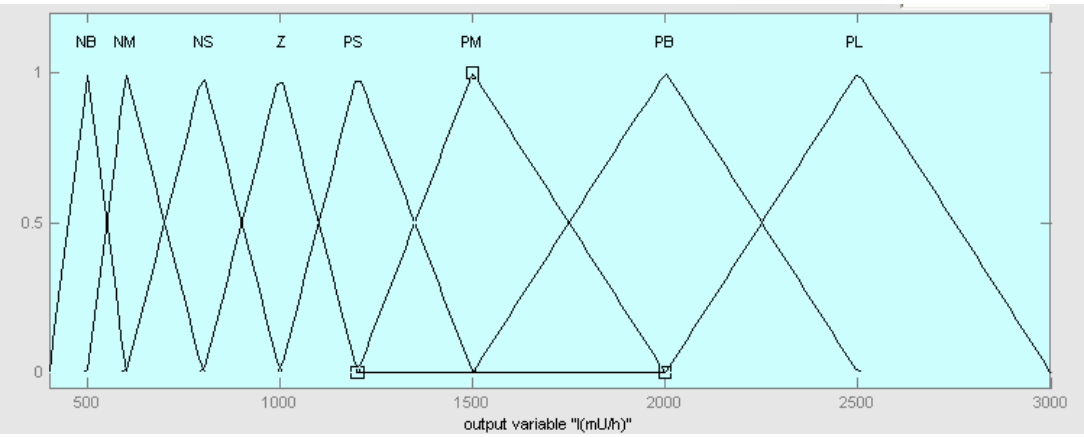
Funcţiile de apartenenţă se aleg conform figurilor:



Funcţiile de apartenenţă ale intrării G(t)



Funcţiile de apartenenţă ale intrării dG/dt



Funcţiile de apartenenţă ale ieşirii I

D.1.3. Baza de reguli

Baza de reguli va fi formată din reguli uniforme de forma:

dacă **G(t)** este **PL** ŞI $\dot{G}(t)$ este **neg**, atunci **I** este **PL** SAU

dacă **G(t)** este **PB** ŞI $\dot{G}(t)$ este **neg**, atunci **I** este **PB** SAU

dacă **G(t)** este **PM** ŞI $\dot{G}(t)$ este **neg**, atunci **I** este **PM** SAU

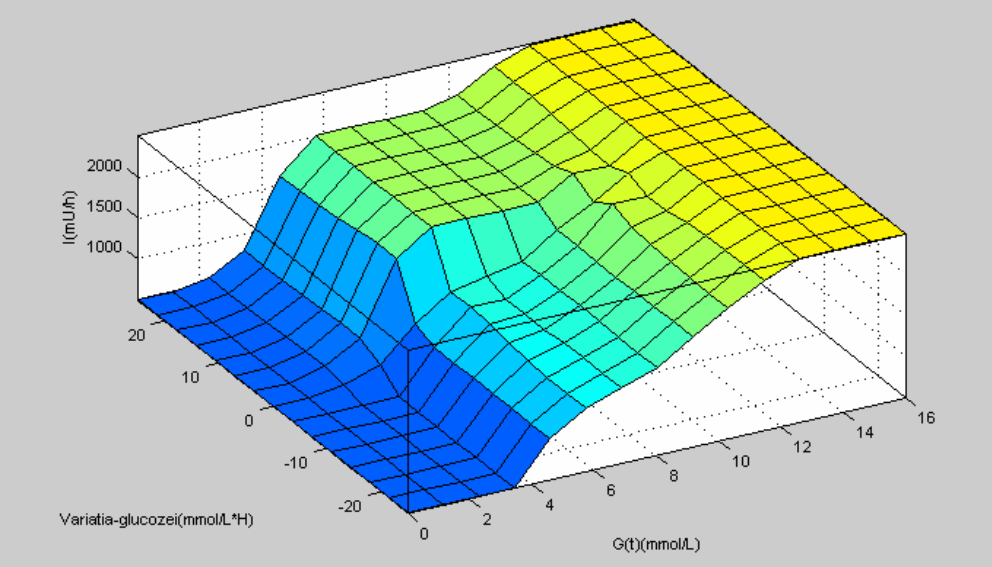
Scrisă sub formă tabelară, baza de reguli arată astfel:

		dG/dt		
		neg	**zero**	**pos**
G(t)	**PL**	PL	PL	PL
	PB	PB	PB	PB
	PM	PM	PM	PB
	PS	Z	PS	PB
	NOM	Z	Z	Z
	NS	NB	NS	Z
	NM	NB	NM	NM
	NB	NB	NB	NB

Determinarea ieşirii acestui regulator cu două intrări şi o ieşire se face folosind inferenţa max-min şi defuzificare cu metoda centrului de greutate(CENTROID).

D.2. Suprafaţa de reglare

Ea prezintă dependenţa mărimii de ieşire de mărimile de intrare. Astfel, suprafaţa de reglare reflectă modul de comportare a regulatorului pentru orice pereche de valori ferme de intrare.



Suprafaţa de reglare

Din figură se observă ca regulatorul fuzzy are un răspuns liniar pe porţiuni fiind astfel superior unor altor tipuri de regulatoare.

E. Intrebări

1. Sunt bazele de reguli continue
2. Sunt bazele de reguli consistente
3. Sunt bazele de reguli complete
4. Daţi un exemplu de îmbunătăţire a bazelor de reguli.

Lucrarea 4
Studiu comparativ între regulatorul PID şi regulatorul fuzzy cu aplicaţie la reglarea nivelului unui vas cu apă

A. Obiective

1) Aprecierea avantajelor/dezavantajelor pe care le oferă fiecare tip de regulator în parte.
2) Construirea unui regulator fuzzy de tip PID.
3) Aprecierea modului de constituire a bazei de reguli.

B. Concepte teoretice ilustrate

Lucrarea de laborator se bazează pe *conceptele teoretice de bază privind structura şi operaţiile unui sistem de reglare cu logică fuzzy*, accentul punându-se în această lucrare pe *compararea performanţelor unui regulator PID cu cele ale unui regulator fuzzy*..

C. Prezentarea aplicaţiei practice

Aplicaţia prezentată în lucrare se bazează pe exemplul demonstrativ din MATLAB sltankrule. In acest exemplu se prezintă o aplicaţie de reglare a nivelului cu ajutorul toolboxului Simulink cu Fuzzy Logic Toolbox. Pentru a porni aplicaţia tastaţi în fereastra de comenzi MATLAB
>> sltankrule
Aplicaţia se referă la reglarea nivelului apei dintr-un vas (TANK) alimentat cu o conductă al cărei debit este reglabil prin intermediul unui robinet de reglare (VALVE). Acumularea apei în vas se realizează cu scurgere liberă. Astfel, debitul de ieşire din vas depinde de nivelul de lichid acumulat în vas (variabil) şi de diametrul conductei de ieşire (constant). Procesul de acumulare a apei în vas prezintă caracteristici puternic neliniare (în sensul că se umple mult mai uşor decât se goleşte, datorită diametrului mic al conductei la ieşire)

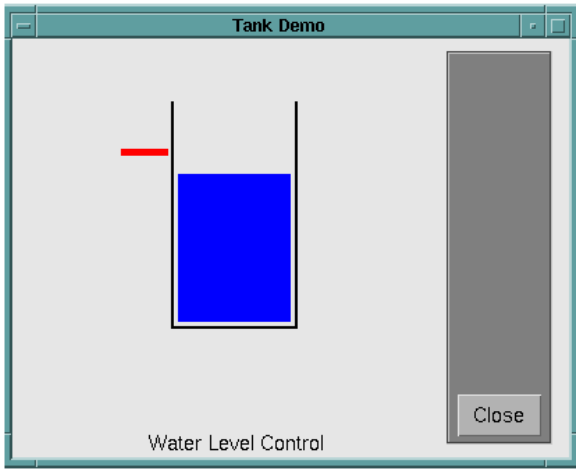


Fig.1. Rezultatul rulării programului sltankrule.

Algoritmul de reglare a nivelului în vas trebuie să ia în calcul nivelul curent al apei şi să comande robinetul de reglare. Asemănător cu un regulator PID, intrarea regulatorului fuzzy va fi eroarea (diferenţa între valoarea dorită şi cea curentă a nivelului). Ieşirea regulatorului va fi viteza cu care se închide/deschide robinetul de reglare situat pe conducta

de intrare. Un prim pas în proiectarea regulatorului fuzzy pentru acest proces poate fi reprezentat de următoarele reguli:

1. *If (level is okay) then (valve is no_change) (1)*
2. *If (level is low) then (valve is open_fast) (1)*
3. *If (level is high) then (valve is close_fast) (1)*

Unul din avantajele utilizării mediului Simulink cu Fuzzy Logic Toolbox este posibilitatea testării performanţelor regulatoarelor fuzzy în buclă închisă (simulând şi procesul modelat ca un bloc Simulink). In figura 2 este prezentat modelul Simulink pentru sistemul de reglare a nivelului cu regulator fuzzy şi regulator PID. Acesta conţine blocul Simulink numit Fuzzy Logic Controller with Ruleviewer. In acelaşi timp, fişierul tank.fis este încărcat în modelul regulatorului FIS.

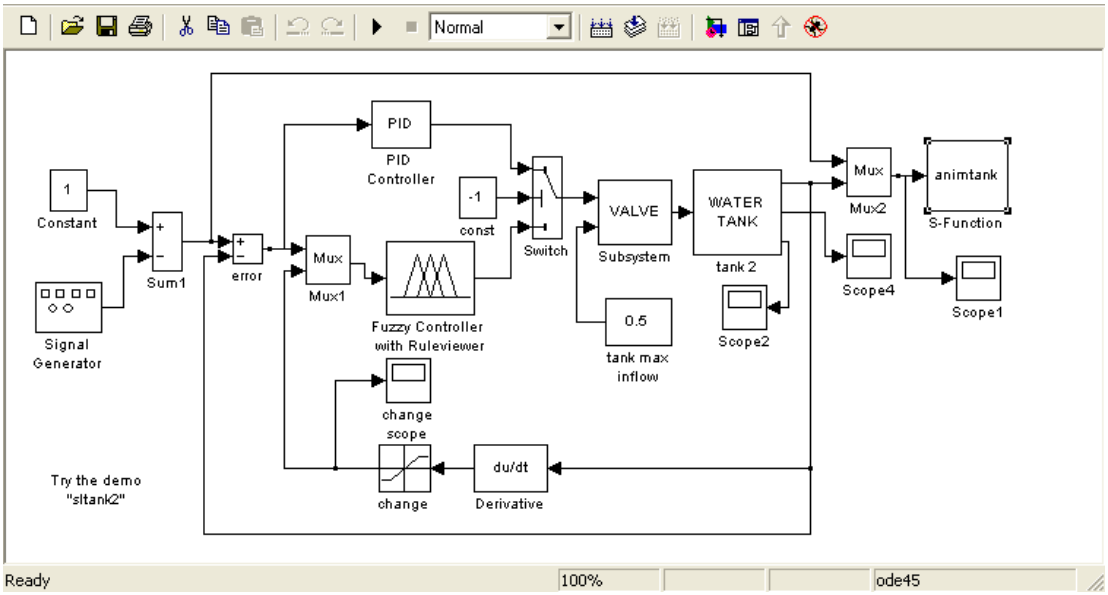


Fig.2. Fisierul simulink sltankrule.

Simulările efectuate cu cele trei reguli propuse arată că acestea nu sunt suficiente şi nivelul tinde să oscileze amortizat în jurul nivelului dorit (Fig.3)

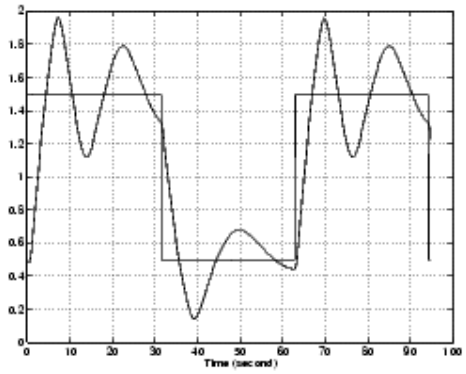


Fig.3. Rezultatele simulărilor utilizând regulator fuzzy cu 3 reguli.

Trebuie să îmbunătăţim regulatorul fuzzy introducând o altă intrare, şi anume viteza de modificare a nivelului (derivata) pentru a modifica lent debitul RR atunci când nivelul se apropie de cel dorit.

4. If (level is good) and (rate is negative), then (valve is close_slow) (1)
5. If (level is good) and (rate is positive), then (valve is open_slow) (1)

Această demonstraţie sltankrule e construită pe baza celor cinci reguli. Ele pot fi vizualizate cu ajutorul FIS Editor tastând în fereastra de comenzi a MATLAB-ului:

>> fuzzy tank

După completarea bazei de reguli se observă comportamentul sistemului de reglare fuzzy la modificarea treaptă a referinţei. Se poate obţine acest lucru prin rularea programului sltankrule cu Start Simulation şi apoi click pe blocul Comparison. In figura 4 sunt prezentate câteva din rezultatele obţinute.

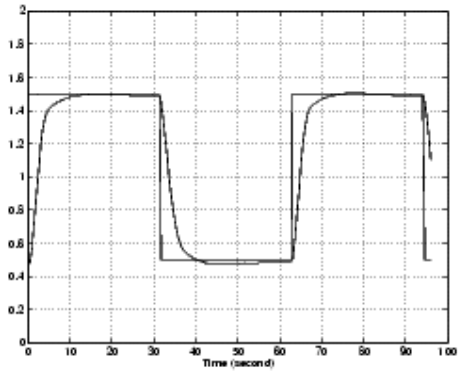


Fig.4. Rezultatele simulărilor utilizând regulator fuzzy cu 5 reguli.

O caracteristică importantă a vasului este faptul că se goleşte mult mai lent decât se umple, din cauza diametrului conductei de ieşire din vas. Proiectarea regulatorului fuzzy ne permite să setăm funcţia de apartenenţă a variabilei lingvistice *close_slow* RR uşor diferită de cea a variabilei lingvistice *open_slow*. Un regulator PID nu permite o facilitate de acest tip pentru procesele neliniare. Comanda RR datorată vitezei de modificare a nivelului de apă (în figura 5 *rate*) la diferite valori ale modificării relative a nivelului (faţă de referinţă, *level*) este reprezentată în figura 5. Aşa cum se observă, există o uşoară asimetrie în suprafaţa de reglare. Aceasta poate fi obţinută tastând *fuzzy tank* urmând apoi paşii descrişi în Lucrarea 1.

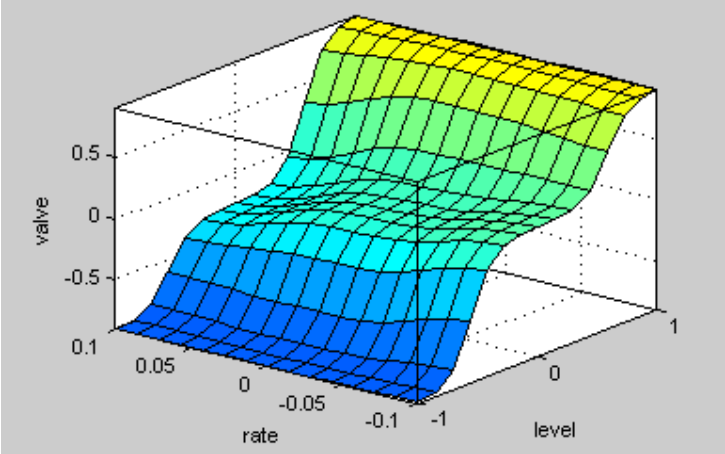


Fig. 5. Suprafaţa de reglare din tank.fis

MATLAB suportă multe toolbox-uri (Control System Toolbox, Neural Network Toolbox, Nonlinear Control Design Blockset ş.a.m.d.) permiţând în consecinţă comparaţii facile ale unui regulator fuzzy cu unul liniar sau chiar bazat pe reţele neuronale.

In această demonstraţie Rule Viewer se deschide în momentul pornirii simulării în Simulink. Acest Rule Viewer este însoţit de animaţie ce explică gradul de îndeplinire a regulilor în timpul simulării (figura 6).

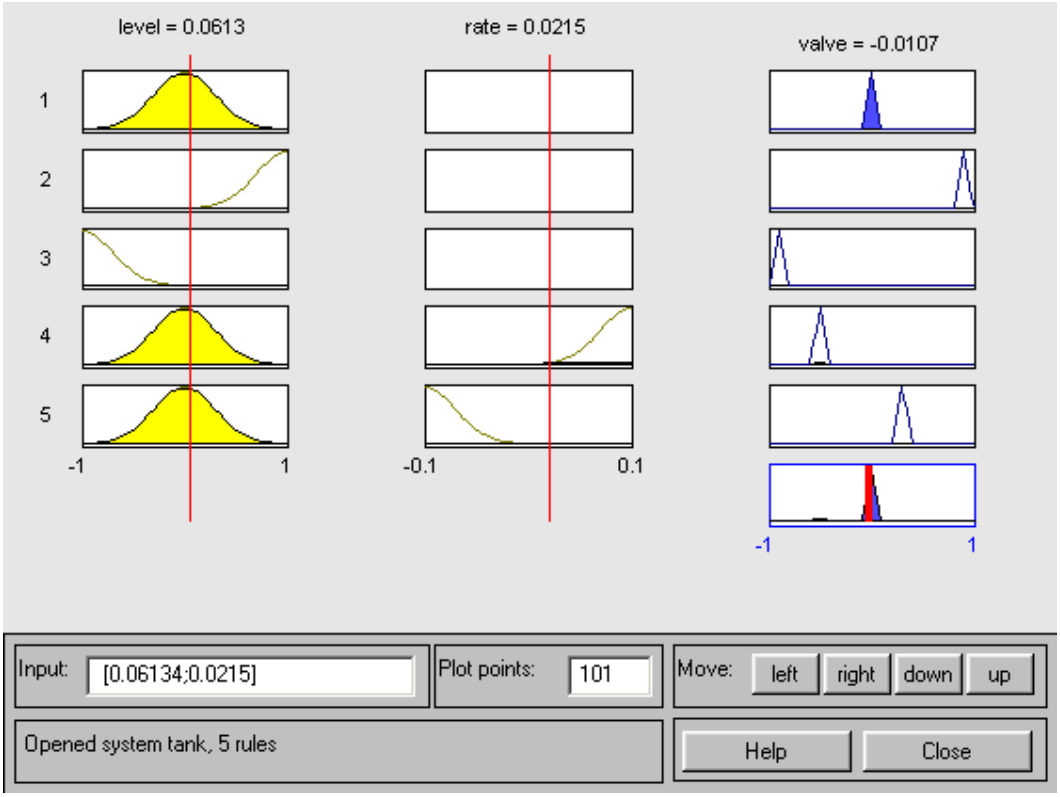


Fig.6. Rule Viewer-ul asociat sltankrule.

Rule Viewer-ul deschis în timpul simulării poate fi utilizat pentru a accesa editorul de funcţii de apartenenţă sau editorul de reguli sau oricare altă interfaţă grafică a toolbox-ului Fuzzy Logic (vezi Lucrarea 1).

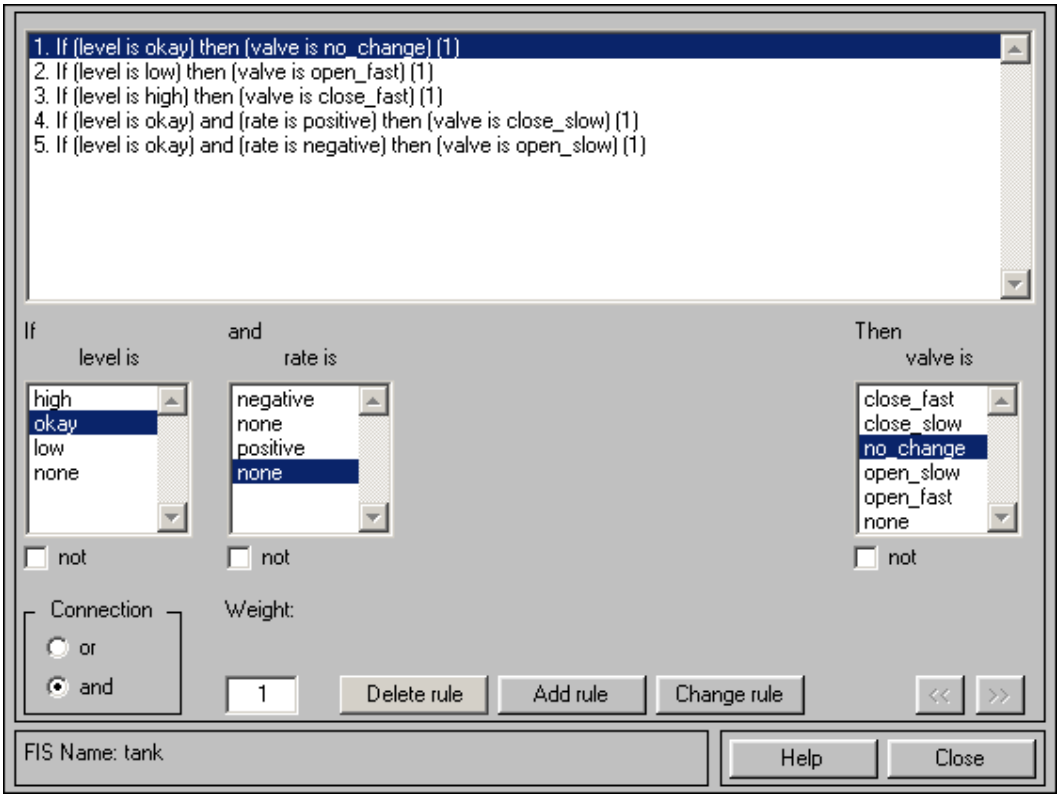


Fig.7 Editorul de reguli asociat regulatorului fuzzy.

De exemplu, dacă se doreşte îmbunătăţirea bazei de reguli, se deschide editorul de reguli cu Edit Rules din meniul *View* al ferestrei deschise Rule Viewer (figura 7). Dar nu înainte de a opri simularea modelului *sltankrule*. Este bine ca modificările în proiectarea regulatorului fuzzy să nu fie făcute în timpul simulării pentru a fi siguri că acestea au fost luate în considerare. De asemenea trebuie salvat în workspace din nou fişierul tank.fis ce conţine modificările algoritmului fuzzy.

Temă: implementarea unui regulator fuzzy de tip PI (ca în curs) şi analiza performanţelor lui pentru procesul dat. Se acordă un bonus pentru acordarea regulatorului astfel construit ca să se potrivească mai bine cu procesul studiat.

Obs. Pentru a exemplifica modul de alcătuire a unei baze de reguli specifice comportării PI – fuzzy considerăm ca variabile de intrare eroarea $e_k = i_k - m_k$ şi variaţia erorii $\Delta e_k = e_k - e_{k-1}$, iar ca variabilă de ieşire variaţia comenzii $\Delta c = c_k - c_{k-1}$. Admitem o reprezentare lingvistică identică pentru cele trei variabile e=Δe=Δc={Negativ Mare, Negativ mic, Zero, Pozitiv mic, Pozitiv Mare} conform tabelului următor.

Δe_k / e_k	**NM**	**Nm**	**Ze**	**Pm**	**PM**
NM	**NM**	**NM**	**NM**	**NM**	**Ze**
Nm	**NM**	**Nm**	**Nm**	**Ze**	**PM**
Ze	**NM**	**Nm**	**Ze**	**Pm**	**PM**
Pm	**NM**	**Ze**	**Pm**	**Pm**	**PM**
PM	**Ze**	**PM**	**PM**	**PM**	**PM**

Lucrarea 5
Implementarea FIS a unor regulatoare fuzzy PD

Scopul final al acestui proiect este realizarea unui regulator de uz general care să poată funcţiona într-o gamă largă de aplicaţii. El va fi testat în urma implementării Simulink pentru diferite procese conduse.

A. Obiective

În cele ce urmează se va realiza un regulator fuzzy de tip PD, atât în varianta Mamdani cât şi în varianta Sugeno.

B. Concepte teoretice ilustrate

Lucrarea de laborator se bazează pe *conceptele teoretice de bază privind structura şi operaţiile unui sistem de reglare cu logică fuzzy*, accentul punându-se în această lucrare pe *aprecierea performanţelor unui regulator fuzzy PD.*

C. Prezentarea aplicaţiei practice

Configuraţia acestui regulatorului propus este următoarea:

- Intrări: eroarea **e**, şi derivata erorii **de**
- Ieşire: comanda **c**

Fuzzificarea va fi realizată după următoarea formulă:
- pentru intrarea **e**, şapte MF: NG, NM, NS , Z , PS , PM şi PG
- pentru intrarea **de**, cinci MF: NG, NS, Z, PS şi PG
- pentru ieşirea **c**: cinci MF - NG, NS, Z, PS şi PG

Etichetele lingvistice utilizate sunt tradiţionale:
NG – negative great/negativ mare
NM – negative medium/negativ mediu
NS – negative small/negativ mic
Z – zero/zero
PS – positive small/pozitiv mic
PM – positive medium/pozitiv mediu
PG – positive great/pozitiv mare.
Se vor utiliza exclusiv variabile normalizate, adaptarea la procesul condus realizându-se prin trei factori de scalare: factorul de scalare pentru eroare **FSe**, factorul de scalare pentru derivata erorii **FSde** şi factorul de scalare pentru comandă **FSc**.

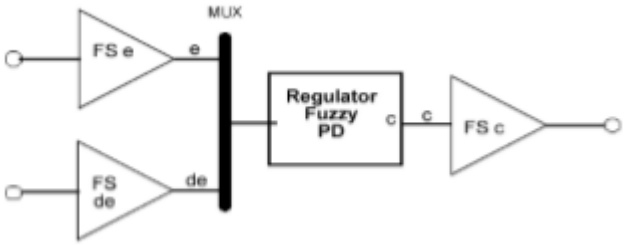


Figura 1. Factorii de scalare ai regulatorului

Scopul final al acestui proiect este realizarea unui regulator de uz general care să poată funcţiona într-o gamă largă de aplicaţii. El va fi testat în urma implementării Simulink pentru diferite procese conduse.

A. Implementarea regulatorului Mamdani

Fuzzificarea variabilelor este prezentată în figura următoare.

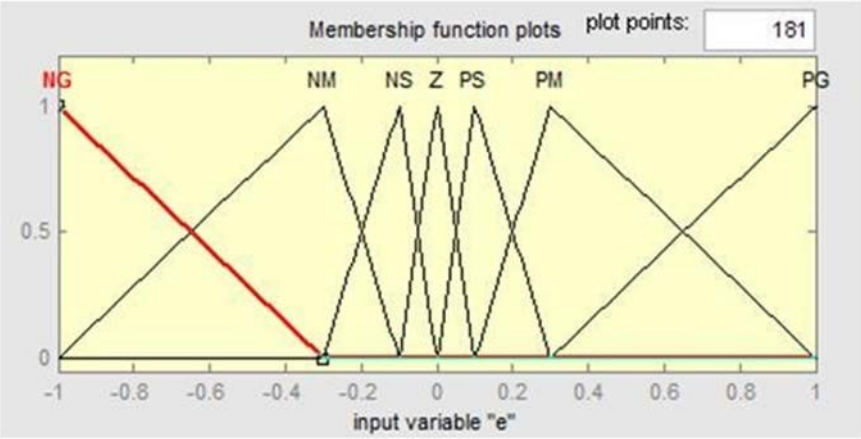


Figura 2. Fuzificarea erorii

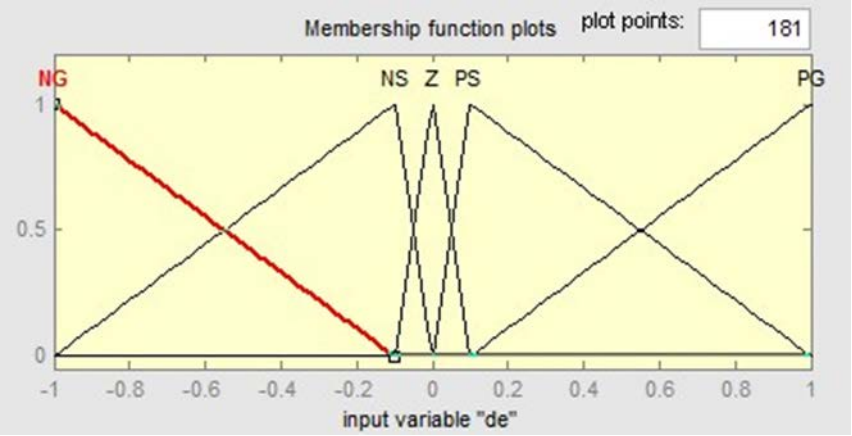


Figura 3 Fuzificarea derivatei erorii

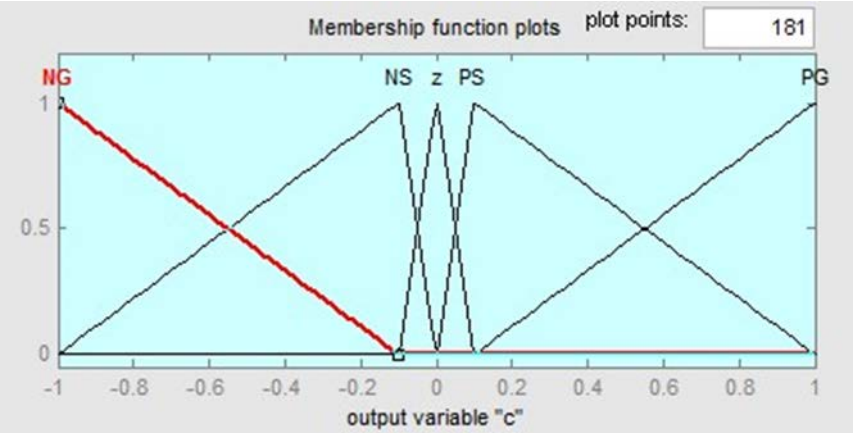


Figura 4. Fuzificarea comenzii

Baza de reguli propusă este următoarea:

1. **If (e is NG) then (c is PG)**
2. **If (e is NM) and (de is NG) then (c is PG)**
3. **If (e is NM) and (de is NS) then (c is PG)**
4. **If (e is NM) and (de is Z) then (c is PS)**
5. **If (e is NM) and (de is PS) then (c is PS)**
6. **If (e is NM) and (de is PG) then (c is Z)**
7. **If (e is NS) and (de is NG) then (c is PG)**
8. **If (e is NS) and (de is NS) then (c is PS)**
9. **If (e is NS) and (de is Z) then (c is PS)**
10. **If (e is NS) and (de is PS) then (c is Z)**
11. **If (e is NS) and (de is PG) then (c is NS)**
12. **If (e is Z) and (de is NG) then (c is PS)**
13. **If (e is Z) and (de is NS) then (c is PS)**
14. **If (e is Z) and (de is Z) then (c is Z)**
15. **If (e is Z) and (de is PS) then (c is NS)**
16. **If (e is Z) and (de is PG) then (c is NS)**
17. **If (e is PS) and (de is NG) then (c is PS)**
18. **If (e is PS) and (de is NS) then (c is Z)**
19. **If (e is PS) and (de is Z) then (c is NS)**
20. **If (e is PS) and (de is PS) then (c is NS)**
21. **If (e is PS) and (de is PG) then (c is NG)**
22. **If (e is PM) and (de is NG) then (c is Z)**
23. **If (e is PM) and (de is NS) then (c is NS)**
24. **If (e is PM) and (de is Z) then (c is NS)**
25. **If (e is PM) and (de is PS) then (c is NG)**
26. **If (e is PM) and (de is PG) then (c is NG)**
27. **If (e is PG) then (c is NG)**

Deşi nu avem decât 27 de reguli, baza de reguli corespunde unui tabel de 35 de reguli, deoarece regulile 1 şi 27 au fost editate cu ajutorul opţiunii **none**: indiferent de valoarea variabilei **de**, starea ieşirii este decisă doar de către **e**.

Suprafaţa de comandă realizată prin defuzzificarea COG este prezentată în Fig. 5. Defuzzificarea MOM produce suprafaţa de comandă din Fig. 6.

Observaţie: Regulile au fost formulate conform logicii lingvistice „dacă eroarea are un anumit semn, atunci comanda trebuie să fie de sens opus”. De exemplu **dacă (e este NG) atunci (c este PG)**. Din acest motiv semnul comenzii **c** trebuie schimbat (se înmulţeşte cu constanta -1).

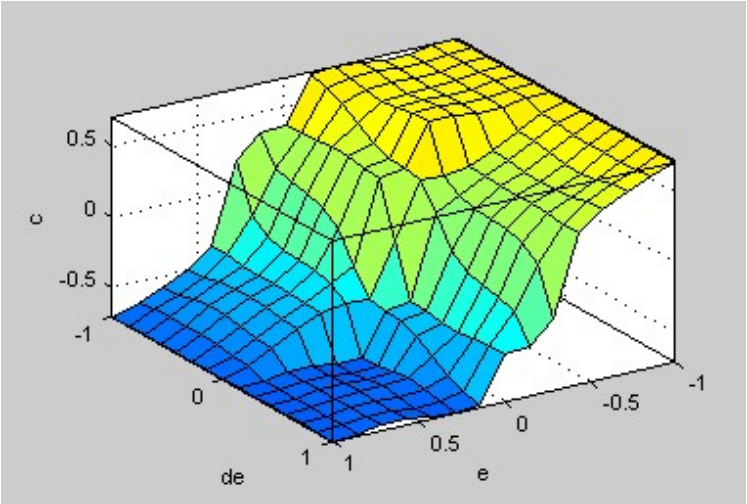


Fig. 5. Suprafaţa de comandă cu defuzzificare COG

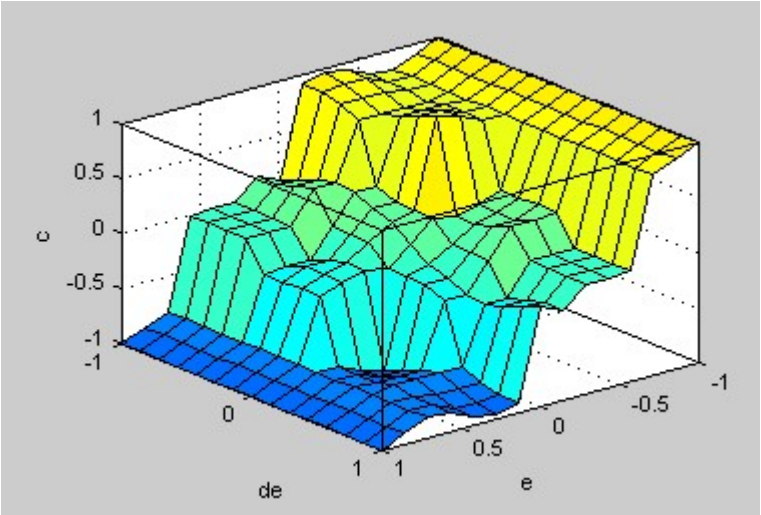


Fig. 6. Suprafaţa de comandă cu defuzzificare MOM

Se remarcă faptul că suprafaţa de comandă generată prin defuzzificarea COG este mai netedă comparativ cu defuzzificarea MOM.

Cu cât mai multe reguli există în baza de reguli, cu atât mai multe puncte de fixare a suprafeţei de comandă avem la dispoziţie, dar pe de altă parte implementarea este mai greoaie.

Implementarea Simulink este prezentată în Fig. 7. Schimbarea semnului comenzii datorată formulării regulilor se realizează prin factorul de scalare FSc. Blocul de saturare care limitează acţiunea regulatorului are limitele ±10.

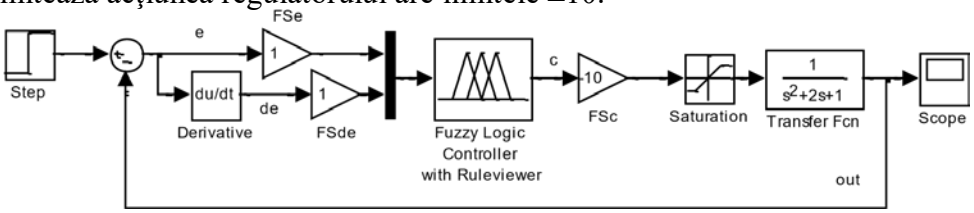


Fig. 7. Implementarea Simulink a regulatorului fuzzy PD

Răspunsul indicial din figura următoare ilustrează comportarea sistemului. Notaţiile folosite sunt: intrarea **u**, ieşirea **out** şi comanda **c**.

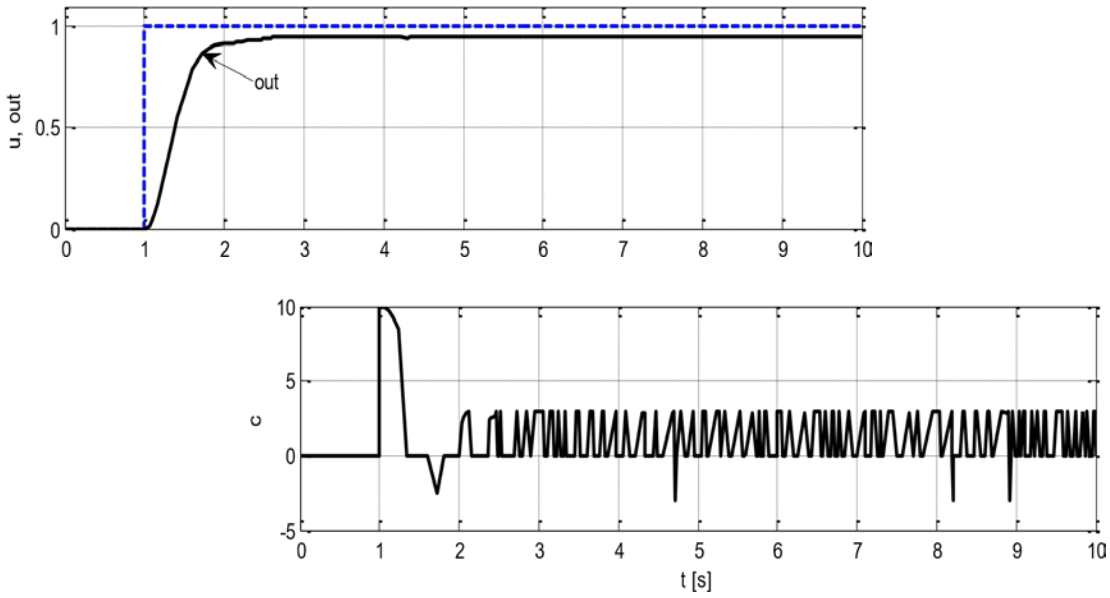


Fig. 8. Răspunsul indicial al regulatorului Mamdani cu baza de reguli 7 x 5

Se remarcă eroarea staţionară precum şi oscilaţiile (chattering-ul) care afectează regimul staţionar, cauzate de lipsa componentei integrale. Pe de altă parte regimul tranzitoriu este ferm, fără suprareglaje sau oscilaţii.

B. Implementarea regulatorului Sugeno

În această secţiune regulatorul Mamdani va fi comparat cu un regulator Sugeno, mai simplu, care diferă doar la fuzzificarea variabilei/variabilelor de ieşire, care se face doar prin singletonuri. Orice controler Mamdani poate fi convertit automat într-un controler Sugeno prin comanda **mam2sug** lansată din fereastra de comandă MATLAB.
În plus reducem baza de reguli la dimensiunea 5 x 5, utilizând numai 5 termeni lingvistici pentru **e**.

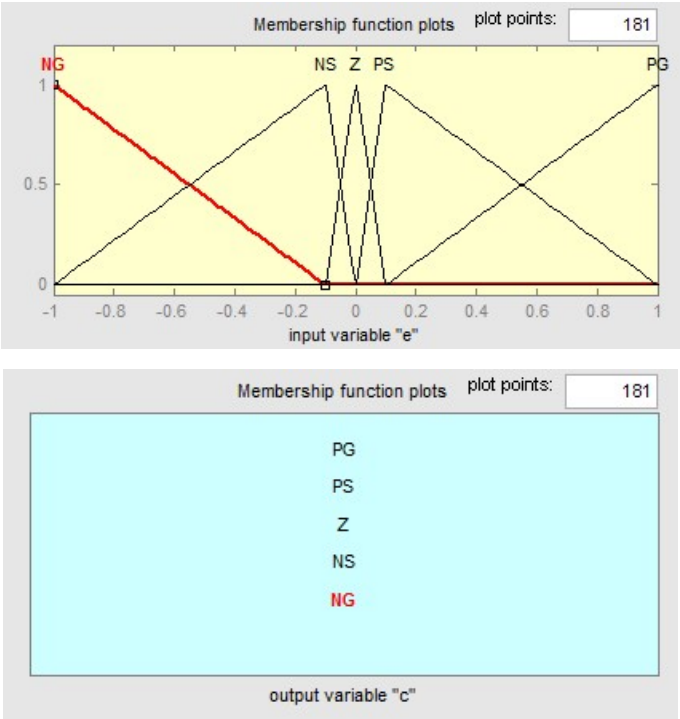


Fig.9. Fuzificarea intrărilor și ieșirilor. **de** e identică cu **e**

1. **If (e is NG) then (c is PG)**
2. **If (e is NS) and (de is NG) then (c is PG)**
3. **If (e is NS) and (de is NS) then (c is PG)**
4. **If (e is NS) and (de is Z) then (c is PS)**
5. **If (e is NS) and (de is PS) then (c is Z)**
6. **If (e is NS) and (de is PG) then (c is NS)**
7. **If (e is Z) and (de is NG) then (c is PG)**
8. **If (e is Z) and (de is NS) then (c is PS)**
9. **If (e is Z) and (de is Z) then (c is Z)**
10. **If (e is Z) and (de is PS) then (c is NS)**
11. **If (e is Z) and (de is PG) then (c is NG)**
12. **If (e is PS) and (de is NG) then (c is PS)**
13. **If (e is PS) and (de is NS) then (c is Z)**
14. **If (e is PS) and (de is Z) then (c is NS)**
15. **If (e is PS) and (de is PS) then (c is NG)**
16. **If (e is PS) and (de is PG) then (c is NG)**
17. **If (e is PG) then (c is NG)**

Fig. 10. Baza de reguli a regulatorului PD Sugeno

Suprafaţa de comandă rezultată este prezentată în figura următoare.

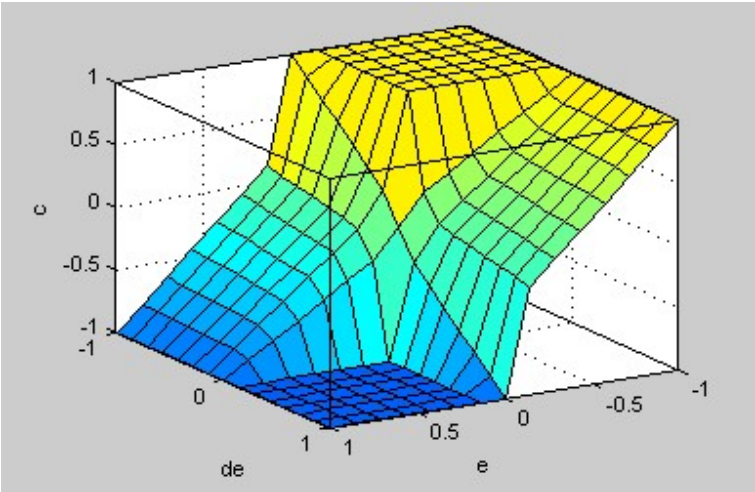


Fig. 11. Suprafaţa de comandă a regulatorului PD Sugeno

Se remarcă o asemănare în linii mari cu suprafaţa de comandă a regulatorului Mamdani care are însă mai multe puncte de ajustare. Răspunsul indicial este foarte asemănător cu cel din Fig. 8, al regulatorului Mamdani cu baza de reguli de dimensiune 7 x 5. Acest fapt justifică utilizarea regulatorului Sugeno în aplicaţiile de reglare în buclă închisă, deoarece este mai simplu şi mai rapid decât regulatorul Mamdani, în timp ce performanţele lor sunt perfect comparabile. Se remarcă lipsa chatteringului, asociată însă cu o eroare staţionară mai mare, comportament cauzat de reducerea bazei de reguli.

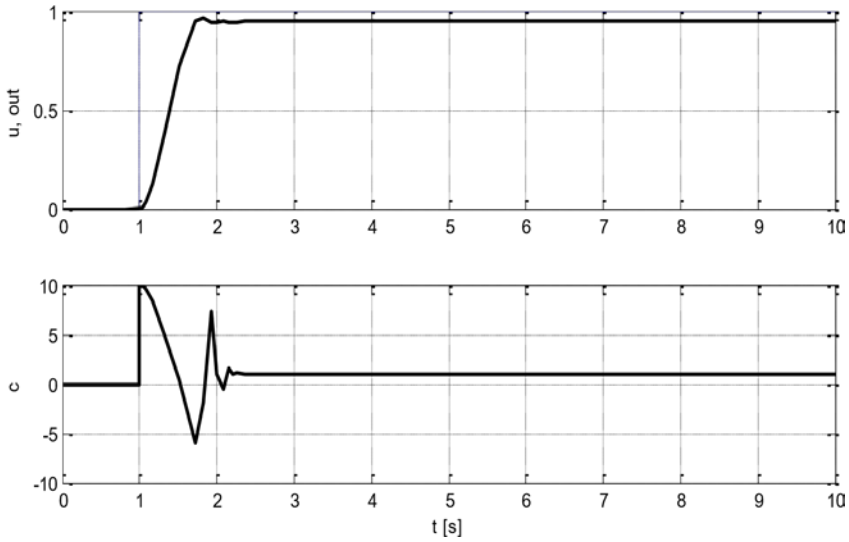


Fig. 12. Răspunsul indicial al regulatorului Sugeno cu baza de reguli 5 x 5

Desfăşurarea lucrării

Studenţii vor reface individual această lucrare.

Lucrarea 6
Implementarea FIS a unor regulatoare fuzzy-interpolative

Scopul final al acestui proiect este realizarea unui regulator de uz general care să poată funcţiona într-o gamă largă de aplicaţii. El va fi testat în urma implementării Simulink pentru diferite procese conduse.

A. Obiective
În cele ce urmează se va realiza un regulator fuzzy-interpolativ

B. Concepte teoretice ilustrate

Lucrarea de laborator se bazează pe *conceptele teoretice de bază privind structura şi operaţiile unui sistem de reglare cu logică fuzzy*, accentul punându-se în această lucrare pe *aprecierea performanţelor unui regulator fuzzy interpolativ*.

C. Prezentarea aplicaţiei practice

Regulatoare fuzzy-interpolative

Sistemele fuzzy-interpolative

În urma executării simulărilor cu fişiere de tip ***.fis** se observă că în general simulările necesită mult timp iar în unele cazuri se produc chiar blocaje, în funcţie şi de configuraţia calculatorului şi de parametrii aleşi pentru simulare (metoda de integrare, erori admisibile, etc.) şi de parametrii sistemului de reglare. O metodă eficace de înlăturare acestui aspect, fără însă a renunţa la avantajul fundamental al sistemelor fuzzy, adică la reprezentarea lingvistică, este apelul la conceptul de *sistem fuzzy-interpolativ* **SFI**.

Un **SFI** este un sistem fuzzy care poate fi direct echivalat cu un *tabel de căutare* **TC** cu interpolare liniară (*look-up-table* **LUT**). Un **TC** este una dintre structurile de date cele mai des întâlnite în ştiinţa calculatoarelor, constând dintr-o arie n–dimensională sau o arie asociativă n–dimensională. Utilizarea **TC** în locul altor algoritmi de calcul al funcţiilor algebrice, ca de exemplu seria Taylor, necesită o capacitate mare de memorie, dar oferă în schimb o accelerare semnificativă a calculului. În esenţă calculul constă din căutarea în tabel a unor valori şi interpolarea lor, fie prin interpolări spline sau polinomiale, fie cel mai simplu, prin interpolări liniare. Deoarece capacitatea memoriilor a crescut continuu, la fel ca şi viteza şi fiabilitatea lor, în timp ce preţul lor este în continuă scădere, utilizarea **TC** este tot mai populară.

În aplicaţiile noastre vom apela la un **TC** bidimensional cu două intrări **e** şi **de** şi o ieşire **c**. În cazul regulatorului Mamdani **TC** poate fi modelat printr-o matrice cu dimensiunea 7 x 5 iar in cazul regulatorului Sugeno printr-o matrice cu dimensiunea 5 x 5. Un astfel de **TC** este disponibil în Simulink sub forma prezentată în Fig. 1.

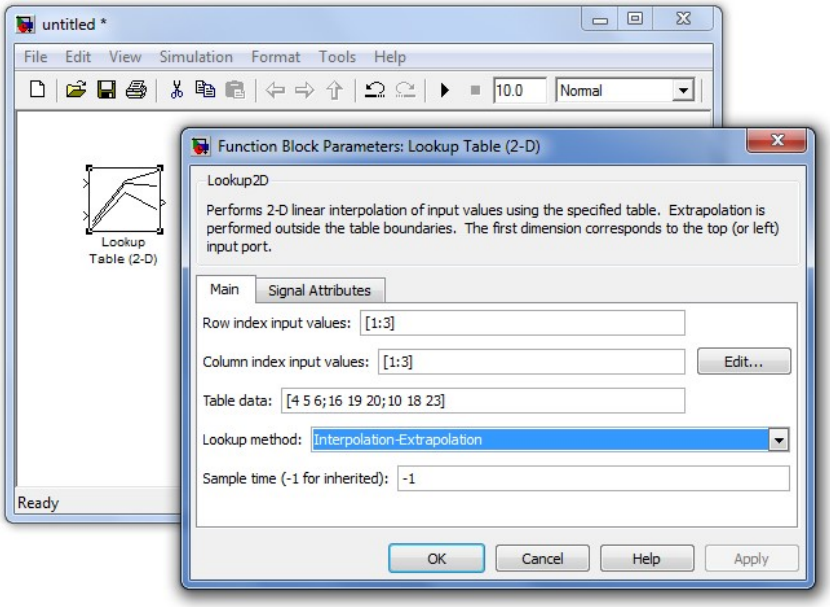


Fig. 1. Tabelul de căutare bidimensional din biblioteca Simulink

Coordonatele nodurilor de interpolare sunt fixate de valorile numerice înscrise în vectorii corespunzători celor două intrări **row** şi **column** iar valorile din tabel **table data** definesc ieşirile tabelului. Valorile vectorilor de intrare trebuie înscrise în ordine crescătoare. Dacă intrările depăşesc domeniul de definiţie calculul ieşirii se poate face fie prin extrapolare fie prin limitare (păstrarea în afara tabelului a valorilor marginale).

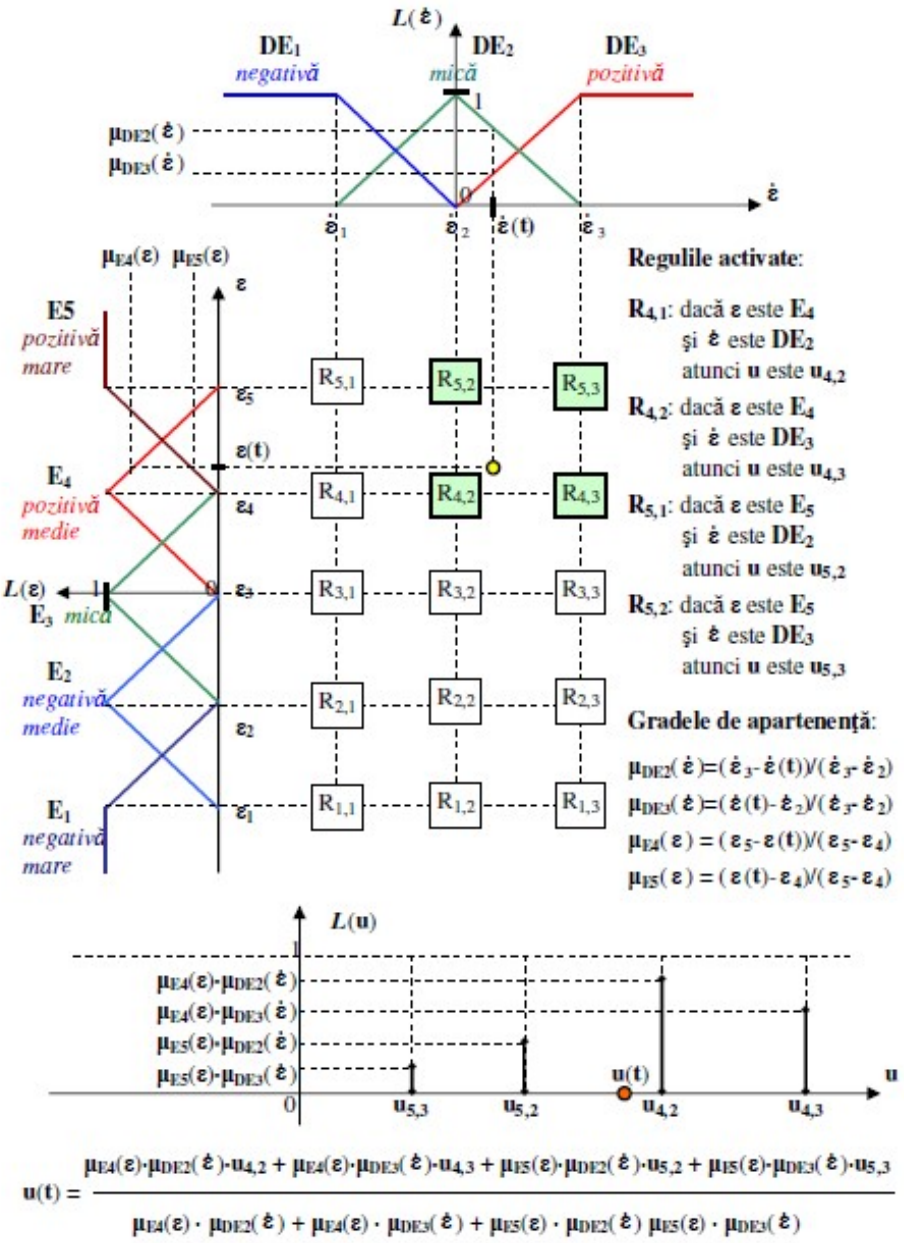


Fig. 2. Ilustrarea funcţionării unui regulator PD fuzzy-interpolativ

Trecerea de la cazul bidimensional **PD** mai uşor de reprezentat mental la cazurile ndimensionale, de exemplu la regulatorul tridimensional de tip **PID**, se poate ilustra prin figura următoare, în care regulatoarele **PD** pot fi asemănate cu paginile unei cărţi.

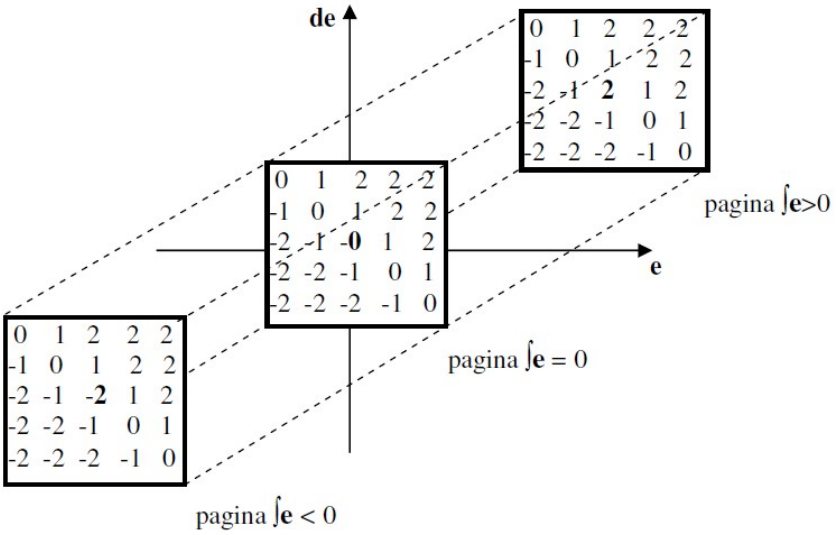


Fig. 3. Regulatorul fuzzy-interpolativ PID

Sintaxa care trebuie respectată în această situaţie este următoarea:

table: **cat(3, [pagina ⌠e<0], [pagina ⌠e=0], [pagina ⌠e>0])**

Implementările fuzzy-interpolative

Implementările fuzzy-interpolative ale regulatoarelor PD Mamdani cu 35 de reguli şi Sugeno cu 25 de reguli prezentate anterior sunt următoarele:

Regulatorul Mamdani:

row (**e**): **[-1 -0.3 -0.1 0 0.1 0.3 1]**

column (**de**): **[-1 -0.1 0 0.1 1]**

table (**c**): **[-1 -1 -1 -1 -1; -1 -1 -0.3 -0.3 0; -1 -1 -0.3 0 0.3; -0.3 -0.3 0 0.3 0.3; -0.3 0 0.3 1 1; 0 0.3 0.3 1 1; 1 1 1 1 1]**

Regulatorul Sugeno:

row (**e**): **[-1 -0.1 0 0.1 1]**

column (**de**): **[-1 -0.1 0 0.1 1]**

table (**c**): **[-1 -1 -1 -1 -1; -1 -1 -0.3 0 0.3; -0.3 -0.3 0 0.3 0.3; -0.3 0 0.3 1 1; 1 1 1 1 1]**

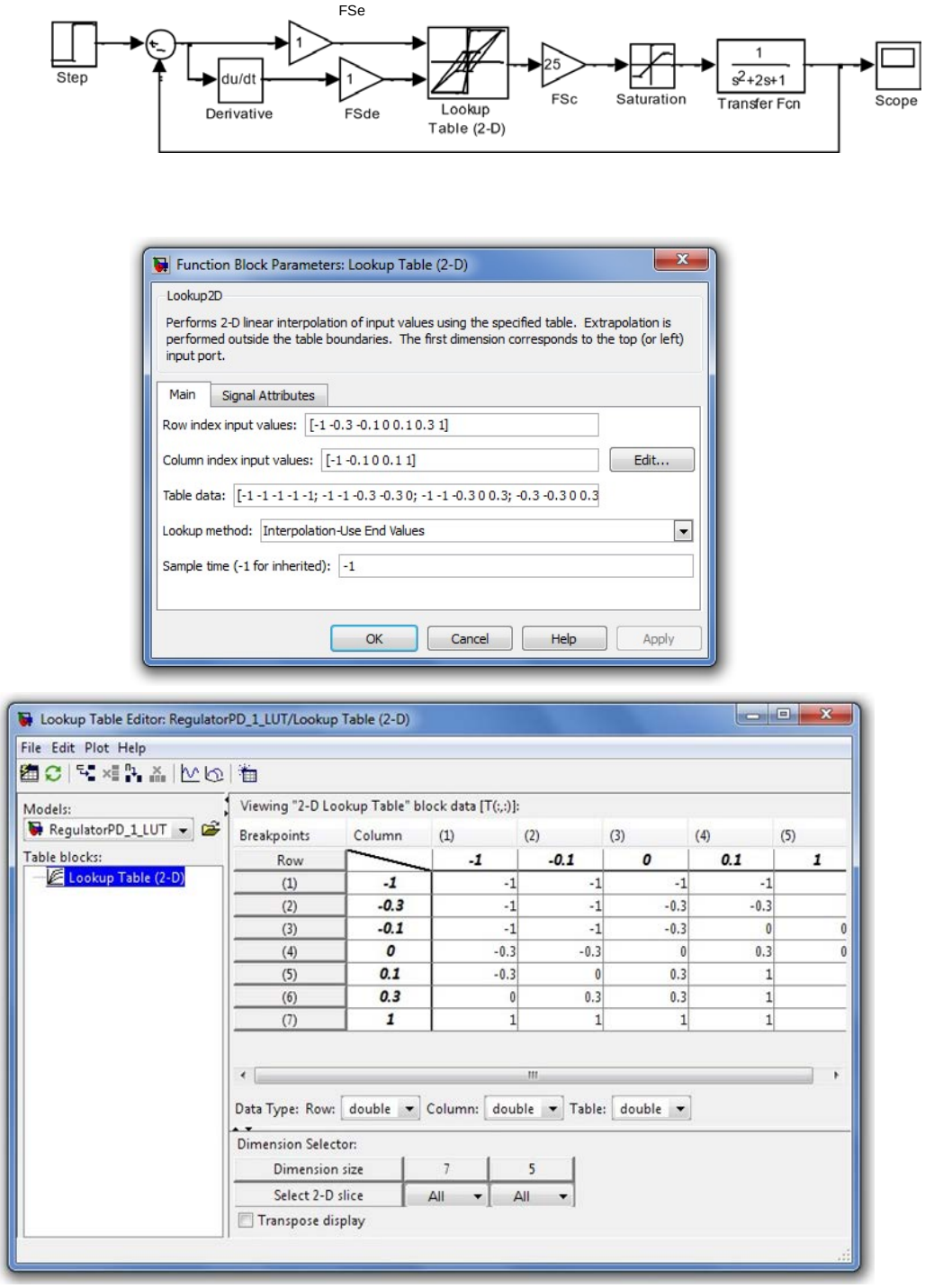


Fig.4. Implementarea Simulink a regulatorului fuzzy interpolativ (7x5)

După implementarea celor două regulatoare se poate constata cu uşurinţă că simulările nu mai ridică absolut nici o problemă computaţională, ele se derulează practic instantaneu, fără riscul blocajelor. Spre deosebire de cazul utilizării fişierului ***.fis** în care pentru anumite valori ale parametrilor sistemului de reglare se produc blocaje, acum putem încerca orice combinaţie de parametri ai sistemului de reglare, de exemplu creşterea factorului de scalare de la ieşire de la 10 la 25. Comportarea regulatoarelor cu 35 şi cu 25 de reguli la un semnal de test reprezentat cu linie întreruptă se poate analiza în figurile următoare.

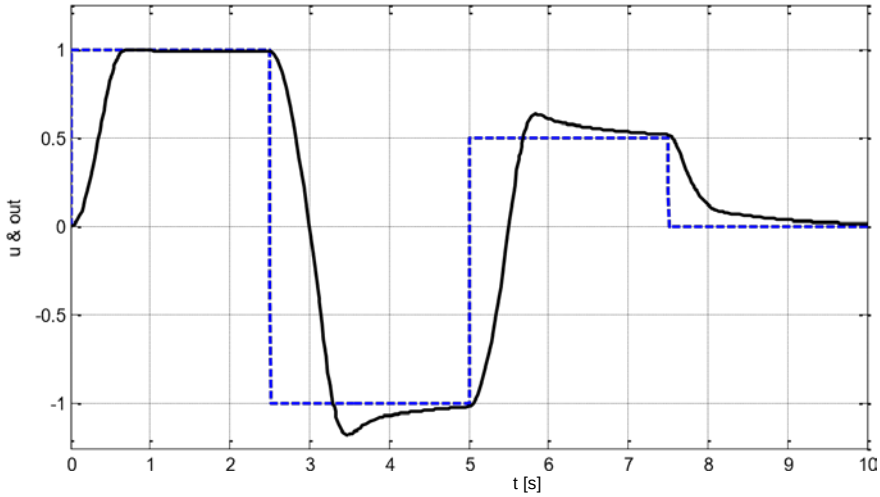


Fig. 5. Un răspuns în timp al regulatorului PD fuzzy cu 35 de reguli

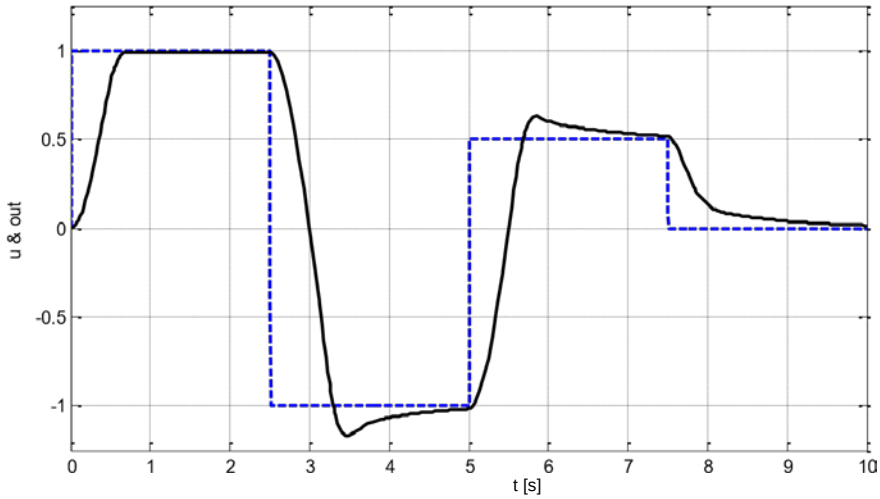


Fig. 6. Un răspuns în timp al regulatorului PD fuzzy cu 25 de reguli

Se remarcă marea asemănare dintre cele două răspunsuri în timp, argument în favoarea utilizării variantei mai simple, cu 25 de reguli. Numărul de reguli în sine este însă mai puţin relevant decât modul în care aceste reguli sunt formulate şi ajustate. După cum se observă, răspunsurile suportă multe îmbunătăţiri.

Diferenţa majoră a regulatoarelor fuzzy faţă de regulatoarele convenţionale este numărul mult mai mare de mecanisme de ajustare care pot influenţa performanţele. Aceasta le oferă un avantaj fundamental, mai ales în aplicaţiile dificile sau despre care nu avem cunoştinţe suficiente pentru a le trata în mod optimal, dar pe de altă parte presupune o ajustare mai laborioasă. Dacă în cazul unui regulator PD liniar sunt doar doi parametri de ajustare, la care putem adăuga factorii de scalare de la intrare, în cazul regulatorului fuzzy-interpolativ putem identifica cel puţin următorii parametri de ajustare: factorul de scalare de la ieşire FSc, fuzzificarea de la intrare care cuprinde 25 de noduri de interpolare şi fuzzificarea de la ieşire cu 5 valori. În cazul unui sistem fuzzy obişnuit mai intervin şi formele funcţiilor de apartenenţă, prin posibilitatea utilizării funcţiilor Gauss, sigmoidale, etc.

În figura următoare se prezintă rezultatele unei ajustări orientate în direcţia reducerii suprareglajelor din Fig. 7, alte obiective fiind la fel de uşor de atins.

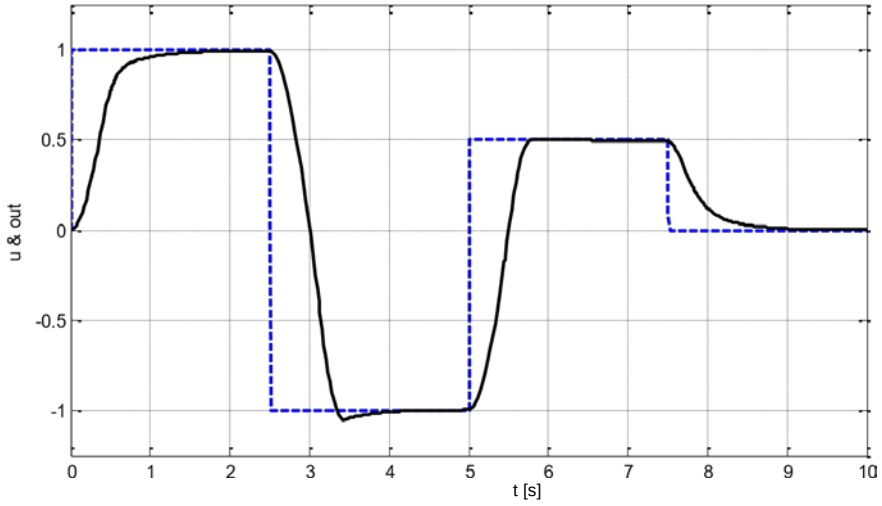


Fig. 7. Un răspuns în timp cu minimizarea suprareglajelor

Desfăşurarea lucrării

Studenţii sunt invitaţi să încerce diferitele mecanisme de ajustare posibile şi efectele lor asupra performanţelor regulatorului fuzzy-interpolativ.

Lucrarea 7
Regulatoare neuro fuzzy implementate in MATLAB ANFIS

Scopul final al acestui proiect este realizarea unui regulator de uz general care să poată funcționa într-o gamă largă de aplicații. El va fi testat în urma implementării Simulink pentru diferite procese conduse.

A. Obiective
   În cele ce urmează se va realiza un regulator neuro fuzzy

B. Concepte teoretice ilustrate

Lucrarea de laborator se bazează pe *conceptele teoretice de bază privind structura și operațiile unui sistem de reglare cu logică fuzzy*, accentul punându-se în această lucrare pe *aprecierea performanțelor unui regulator fuzzy neuro-fuzzy.*

C. Prezentarea aplicației practice

Introducere în rețelele neuronale

Mulțimea de posibilități de ajustare a controlerelor fuzzy creează complicații, care se pot evita prin *automatizarea* elaborării relației fuzzy. Sinteza automată sau chiar autosinteza sistemelor de comandă poate fi obținută prin metode ale inteligenței artificiale. Cea mai promițătoare abordare constă din fuzionarea logicii fuzzy cu *rețelele neuronale* **RN**. În această abordare mulțimile fuzzy sunt utilizate în reprezentarea cunoștințelor, în timp ce rețelele neuronale sintetizează regulile fuzzy *prin învățare pe bază de exemple*. Controlerele neuro-fuzzy astfel obținute sunt capabile de a rezolva unele dintre cele mai complicate sarcini, cum ar fi recunoașterea imaginilor, fără intervenția substanțială a omului.

RN sunt alcătuite din mulțimi de elemente de prelucrare a informației - neuroni - interconectați într-o rețea. Un neuron **i** are **n** intrări, x_1^i, x_2^i, ... , x_n^i și o ieșire y^i, fiecare aparținând mulțimii numerelor reale. O legătură ieșire-intrare dintre nodurile rețelei este reprezentată prin *ponderea sinaptică*, valoare reală care ponderează efectul semnalului de intrare în neuronul receptor. Neuronul va fi afectat de **n** ponderi sinaptice: w_1^i, w_2^i, ... , w_n^i. Ponderile pot fi excitatoare dacă $w^i > 0$ sau inhibatoare dacă $w^i < 0$. *Activarea* neuronului se definește prin suma ponderată a semnalelor de intrare:

$$S = \sum_{j=1}^{n} w_j^i x_j^i$$

Ieșirea neuronului va fi dată de *funcția neuronală* (de ieșire) $y^i = f(s)$, care poate avea diferite forme: treaptă, rampă, funcții sigmoidale, etc.

Neuronii sunt aranjați în *straturi*, capacitatea de reprezentare a rețelei depinzând de mărimea și de numărul straturilor. Dacă funcțiile neuronale sunt norme **t** sau **s** se obțin neuroni logici de tip **AND** și **OR**. Cu aceștia se pot construi rețele multistrat capabile să modeleze o mare varietate de funcții logice. O astfel de structură (denumită și *procesor logic*), rezultată prin analogie cu logica booleeană a fost propusă de K. **Hirota** și W. **Pedrycz** în 1991. Ea cuprinde trei straturi de neuroni (de intrare, de ieșire și stratul ascuns), fiecare strat conținând neuroni de același tip.

Controlerul neuro-fuzzy poate fi grefat pe procesorul logic neuronal, fiecare variabilă de intrare reprezentând gradul de apartenență a unei mărimi de intrare la TL corespunzător din CC aferent mărimii respective. Prin topologia lor, *sinapsele codifică regulile de conducere de tip **dacă** ...**atunci**.* În cazul procesorului logic de mai sus, fiecare nod din stratul intermediar poate fi considerat ca și corespunzând unei clauze, fiind analog unui minitermen din cazul exprimării analitice prin forma canonică disjunctivă a funcțiilor logice. În cazul general cunoștințele sunt "împărțite" între neuroni și sinapse, astfel că este cvasi-imposibilă exprimarea inteligibilă a relației fuzzy conținută în rețea.

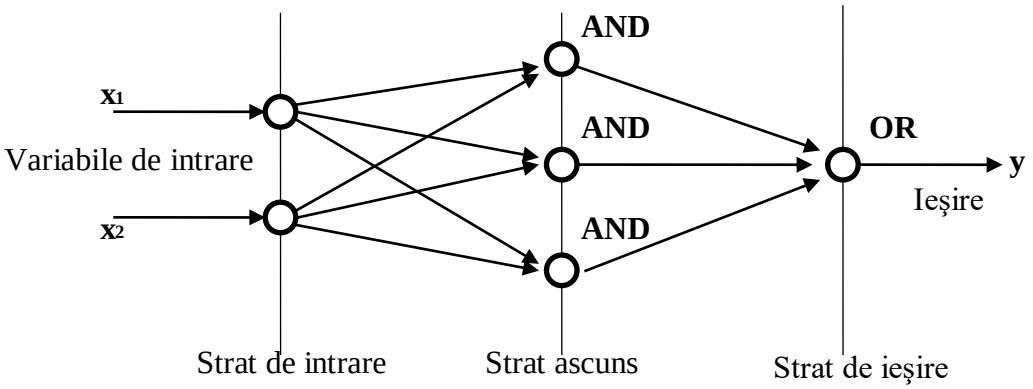


Fig.1 Procesor logic cu rețea neuronală

RN sunt capabile de a învăța pe bază de exemple. Exemplele constau din seturi de mărimi de intrare și mărimile de ieșire pe care dorim să le obținem în cazul acelor intrări. Învățarea este de tip parametric, constând din ajustarea ponderilor sinaptice, astfel încât un anumit indice de performanță (de exemplu distanța euclidiană) să fie optimizat. Dintre parametrii algoritmilor de învățare, esențiali sunt *convergența*, fără de care nu sar obține soluții și *factorul de învățare* indicând viteza de soluționare.

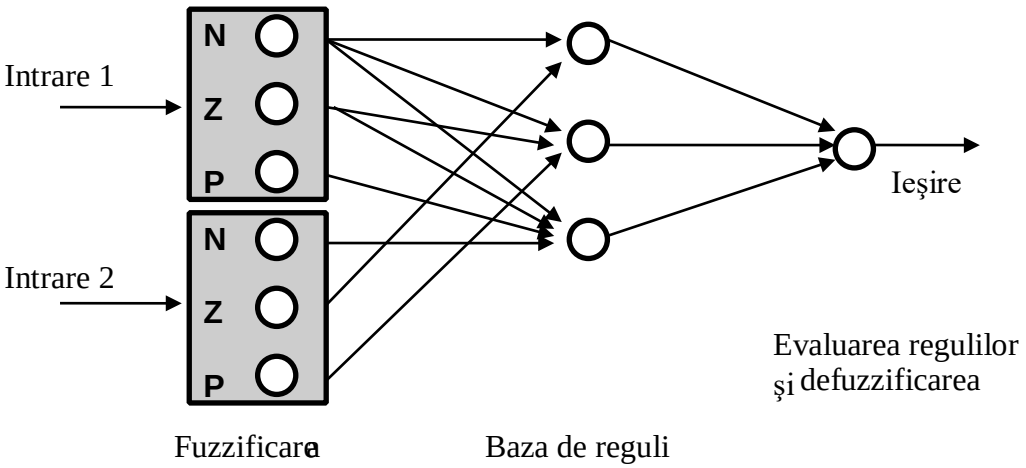


Fig. 2 Controler neuro-fuzzy

Descrierea toolkitului ANFIS

Tool-kit-ul destinat implementării sistemelor neuro-fuzzy în Matlab se numește ANFIS (*adaptive neuro-fuzzy inference system*). Fiind dat un set de date intrare-ieșire de antrenare, ANFIS generează un fișier de tip ***.fis** având parametrii astfel ajustați încât să producă un răspuns identic cu datele de antrenare. Învățarea se realizează fie printr-un algoritm de tip *backpropagation* fie printr-o variantă a metodei celor mai mici pătrate.

Metodele de tip backpropagation urmează următoarele etape:

1. Se introduce în **RN** setul de date de antrenare.
2. Se compară ieșirile calculate cu setul de date de antrenare. Se calculează erorile pentru fiecare dintre neuronii de ieșire.
3. Pentru fiecare neuron de ieșire se calculează o valoare și un factor de scalare, care ar produce ieșirea dorită pentru întreaga rețea. Pe baza aceasta se calculează erorile locale la nivelul fiecărui neuron de ieșire.
4. Se ajustează ponderile fiecărui neuron în sensul scăderii erorilor locale.
5. Neuronii din stratul precedent care produc cele mai mari erori locale sunt ponderați în sensul scăderii ponderii lor.
6. Se repetă etapele 3, 4 și 5 asupra neuronilor din stratul anterior.

Tool-kit-ul ANFIS are un caracter preponderant didactic, prezentând o serie de limitări:

• Se aplică numai sistemelor Sugeno cu funcții de apartenență liniare sau singleton;

• Se aplică numai sistemelor cu o singură ieșire;

• Nu suportă funcții de apartenență, norme **t-s** sau defuzzyficări definite de utilizator;

Lansarea toolkit-ului se face prin comanda **anfisedit**, obținându-se interfața grafică din figura 3. Rolul principalelor elemente de acționare este prezentat în figură.

Operațiile pe care interfața ANFIS le poate executa sunt următoarele:

- Încărcarea datelor de antrenare, testare și verificare, sub controlul comenzilor din zona **Load Data**.
- Generarea sau încărcarea unui model inițial FIS cu comenzile din zona **Generate FIS**.
- Vizualizarea modelului FIS (atât cel inițial cât și cel rezultat în urma antrenării rețelei prin butonul **Structure**.
- Alegerea parametrilor metodei de optimizare (backpropagation sau hibridă).
- Alegerea numărului de iterații de antrenare (*traning epochs*) și eroarea de antrenare.
- Antrenarea rețelei neuronale, comandată prin butonul **Train Now**. Procesul iterativ de învățare automată ajustează parametrii FIS-ului. Se afișează erorile fiecărei iterații.

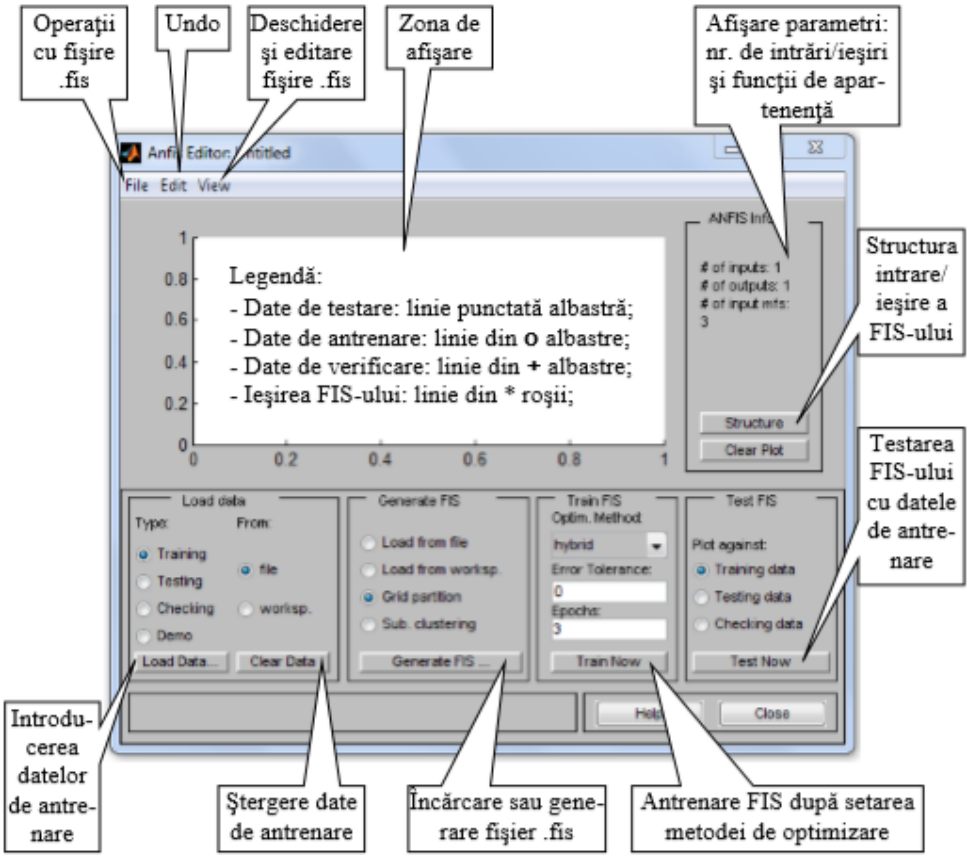


Fig.3. Interfața grafică ANFIS

- Ieșirea FIS-ului este vizualizată și comparată cu datele de antrenare, verificare și testare, după acționarea butonului **Test Now**.

Prin intermediul barei de meniu ANFIS se pot încărca sau salva fișierele FIS, se poate deschide un nou controler Sugeno sau accesa alte interfețe grafice din componența Matlab.

Implementarea ANFIS a unui regulator PD Sugeno

În continuare se va exemplifica optimizarea regulatorului PD Sugeno descris în cu ajutorul ANFIS. În urma încărcării fișierului .fis corespunzător, se obține rețeaua neuronală din figura 4.

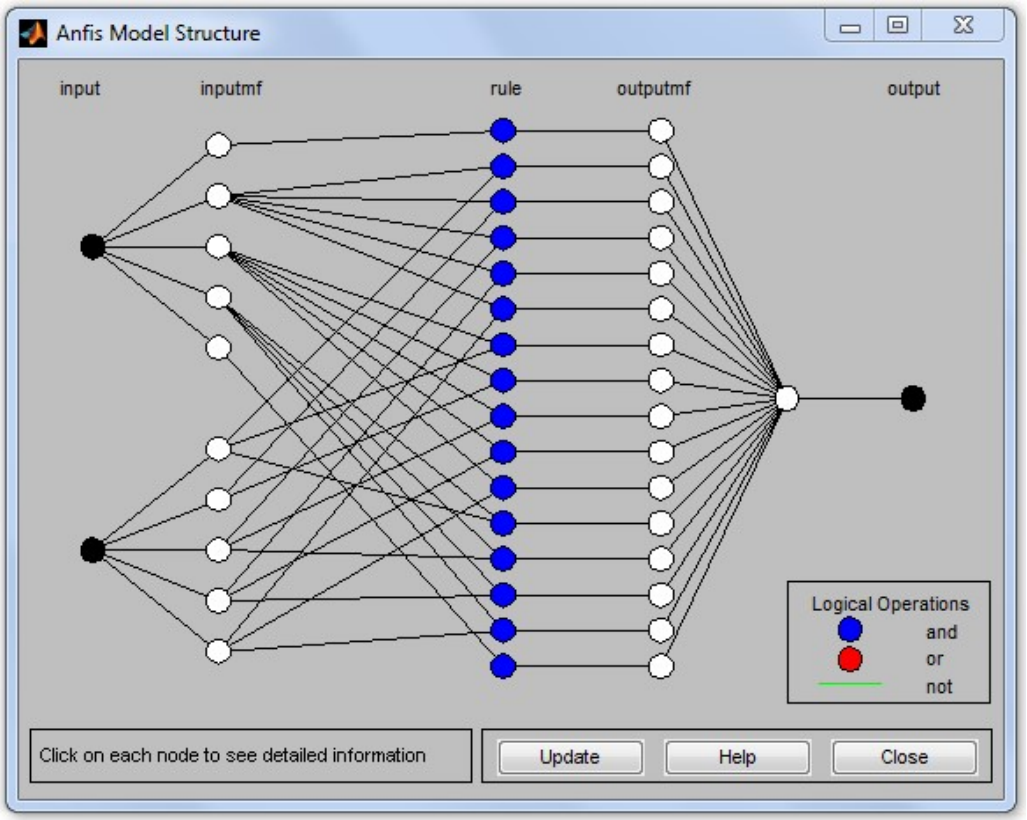


Fig. 4. Structura RN corespunzătoare regulatorului PD Sugeno

Corespondența dintre RN și controlerul Sugeno este evidentă:

- intrările **input** corespund erorii **e** și derivatei sale **de**.

- stratul **inputmf** corespunde termenilor lingvistici rezultați prin fuzzificare.

- stratul **rule - outputmf** asociază câte un neuron fiecărei reguli de conducere.

- ieșirea **output** corespunde ieșirii controlerului Sugeno.

Principala problemă care se ridică în fața proiectării unei aplicații ANFIS este alegerea seturilor de date de antrenare, testare și validare.
În vederea optimizării regulatorului fuzzy PD s-a realizat un set de date de antrenare prin prelevarea datelor obținute din simularea funcționării sistemului de reglare pentru cazul răspunsului indicial. Datele au fost prelevate chiar de la intrările și ieșirea regulatorului, deoarece ANFIS generează doar fișierul **.fis**, el neputând fi extins la întreaga aplicație. A rezultat un fișier de date cu dimensiunea 25 x 3. Parametrii simulării generatoare de date de antrenare pot fi astfel aleși încât să reprezinte o îmbunătățire a funcționării, care nu ar fi putut fi obținută prin ajustarea manuală. Convergența învățării și parametrii ANFIS-ului se observă în fig. 5.

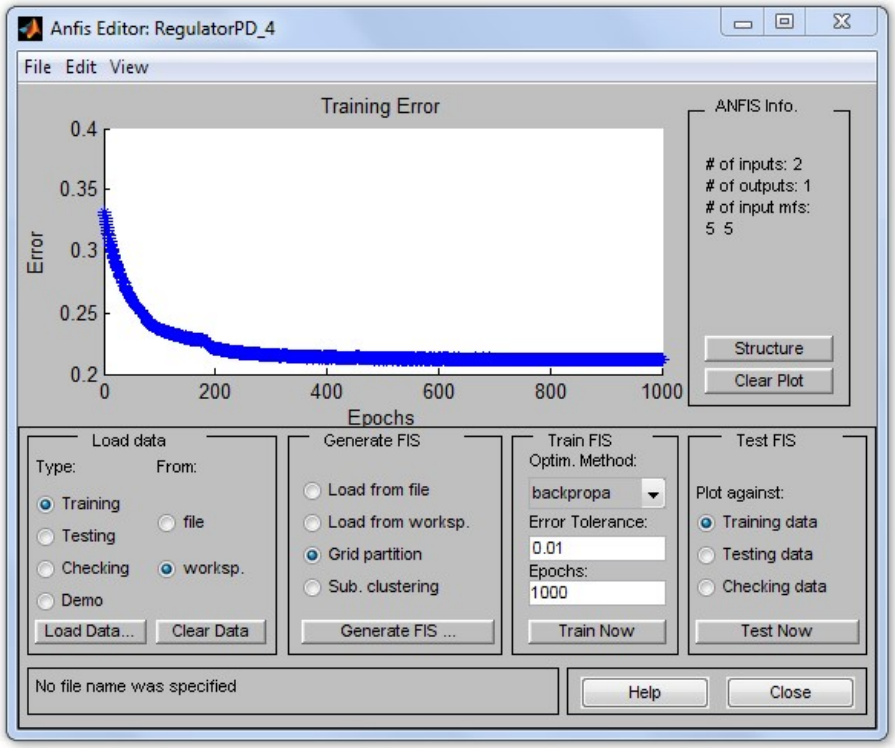


Fig. 5. Convergența învățării pentru un set de date reprezentând un răspuns indicial

Din păcate regulatorul astfel obținut produce un răspuns indicial de slabă calitate, cu o tendință înspre suprareglaj și oscilații pe durata regimurilor tranzitorii. Așa cum se va observa în continuare, principala cale de îmbunătățire a calității sistemelor neuro-fuzzy este alegerea unui set de date de antrenare cât mai adecvat.

Sinteza unui regulator neuro-fuzzy

Utilitatea RN stă în sinteza automată a sistemelor și/sau optimizarea lor. În continuare se ilustrează sintetizarea unui regulator PD neuro-fuzzy. Se pornește de la ideea obținerii unui set de date de antrenare bogat, care să cuprindă o comportare mai nuanțată a regulatorului PD, respectiv un semnal de intrare mai complex, care pe lângă răspunsul indicial să mai cuprindă și alte elemente.

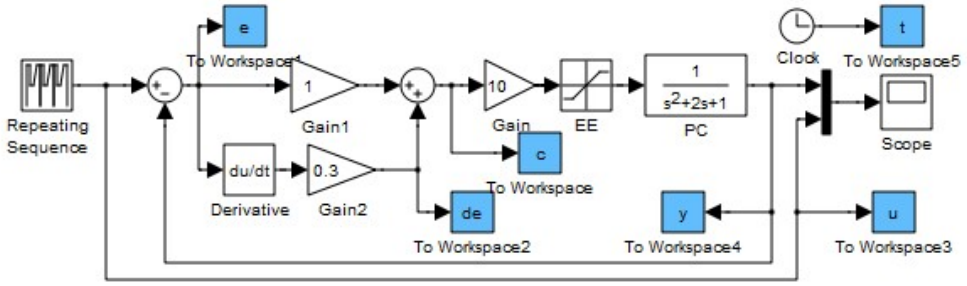


Fig. 6. Sistem de reglare cu regulator PD liniar pentru generarea datelor de antrenare

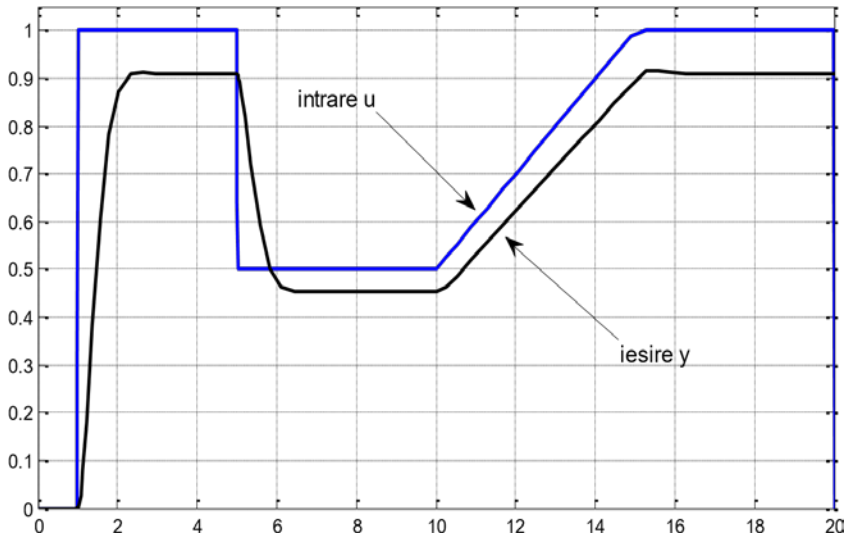


Fig. 7. Răspunsul sistemului de reglare cu regulator PD liniar

Fișierul de date de antrenare obținut are dimensiunea 123 x 3. Reprezentarea grafică a datelor de antrenare apare în fig.8.

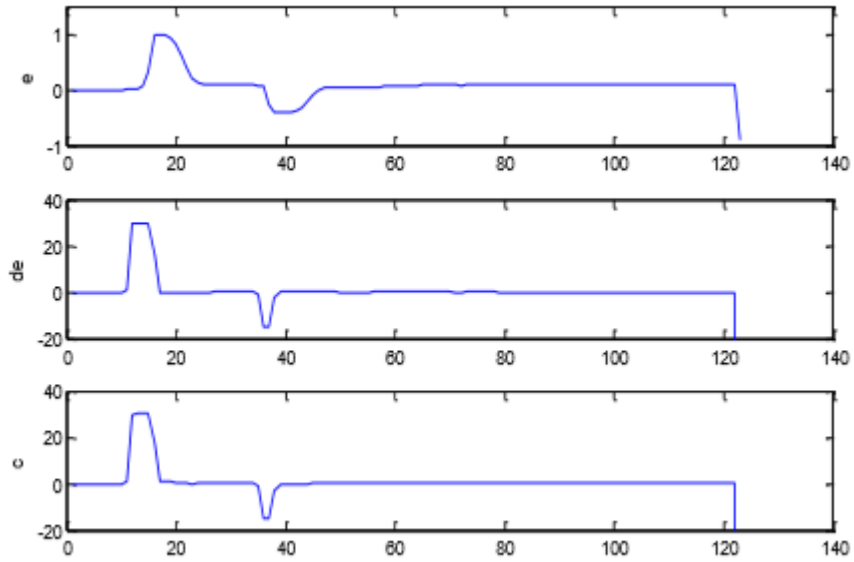


Fig. 8. Datele de antrenare

În urma încărcării datelor de antrenare și a alegerii variantei în care ANFIS va genera un FIS cu două intrări având 7 respectiv 5 Termeni Lingvistici (o bază de 7 x 5 reguli), se obține antrenarea convergentă din fig. 9.

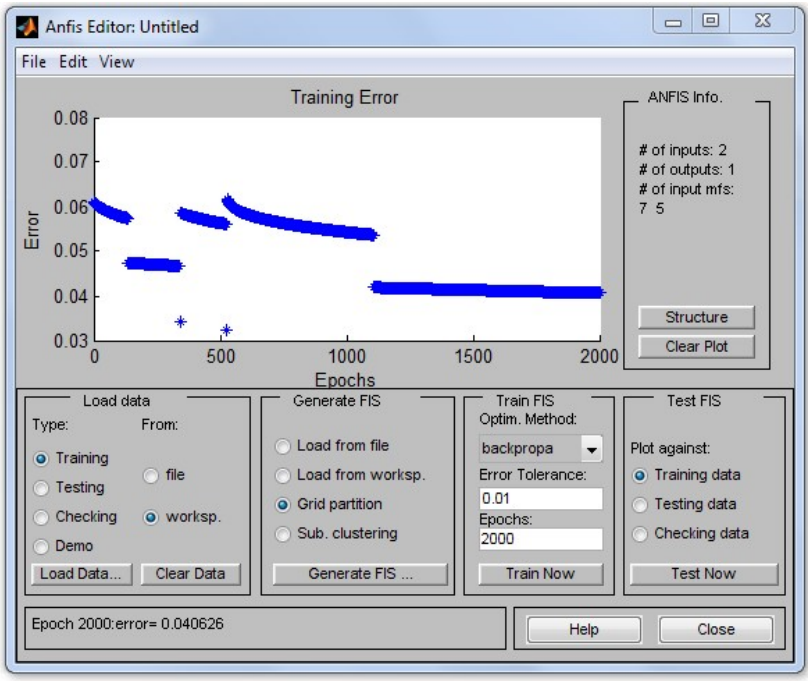


Fig. 9 Antrenarea în cazul unei baze de reguli 7 x 5 (**linear**)

Regulatorul PD Sugeno generat de RN este prezentat în figura următoare. Se remarcă variabila sa de ieșire alcătuită din 35 de valori diferite (singleton), fiecare dintre ele fiind optimizată în cursul antrenării. În acest caz a fost aleasă opțiunea **linear** pentru

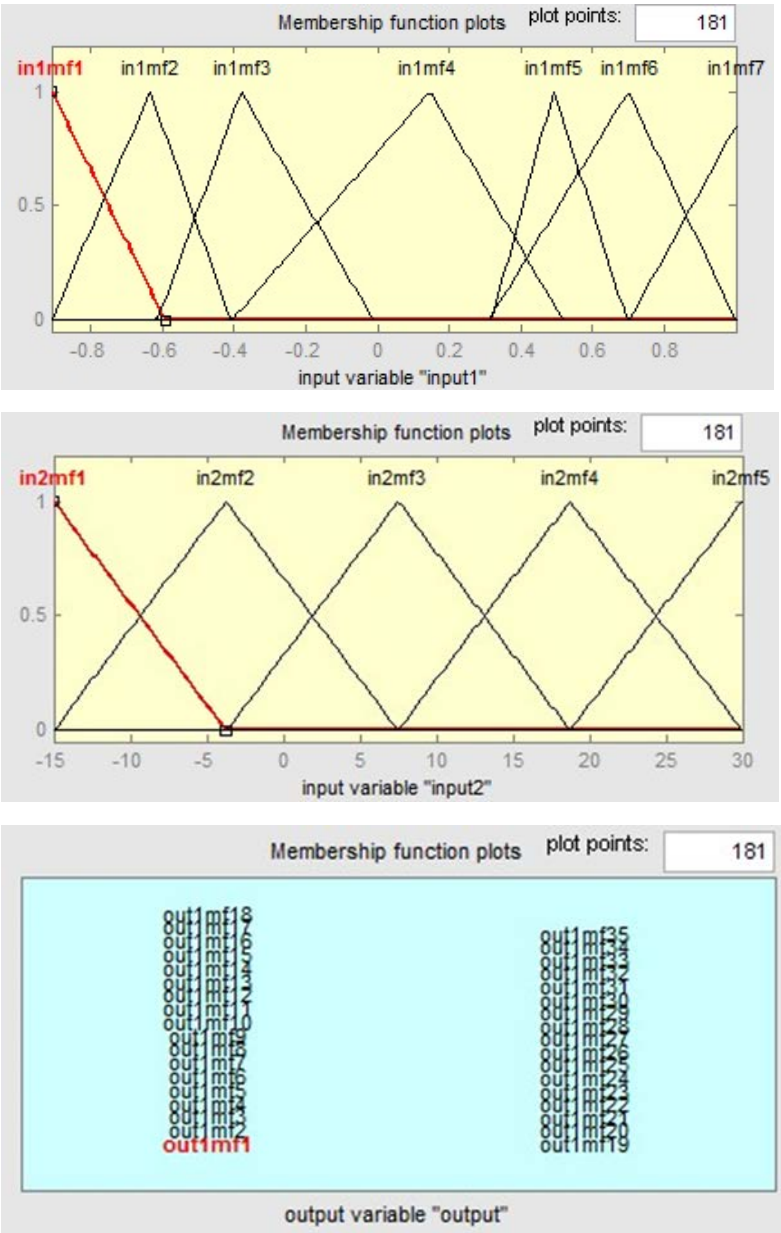


Fig.10. Regulatorul neuro-fuzzy cu baza de reguli 7x5

În cazul alegerii opțiunii **constant** convergența învățării este chiar mai bună (vezi fig. 11), ceea ce nu crește însă în mod necesar calitatea regulatorului.

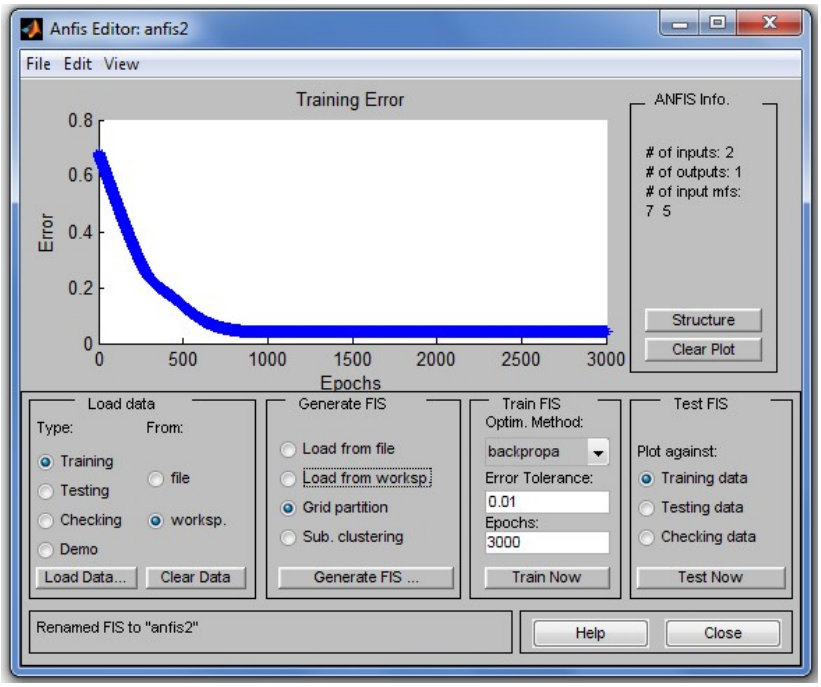


Fig. 11. Antrenarea sistemului neuro-fuzzy cu baza de reguli 7 x 5 (**constant**)

Răspunsul produs de către regulatorul astfel obținut este prezentat în fig. 12.

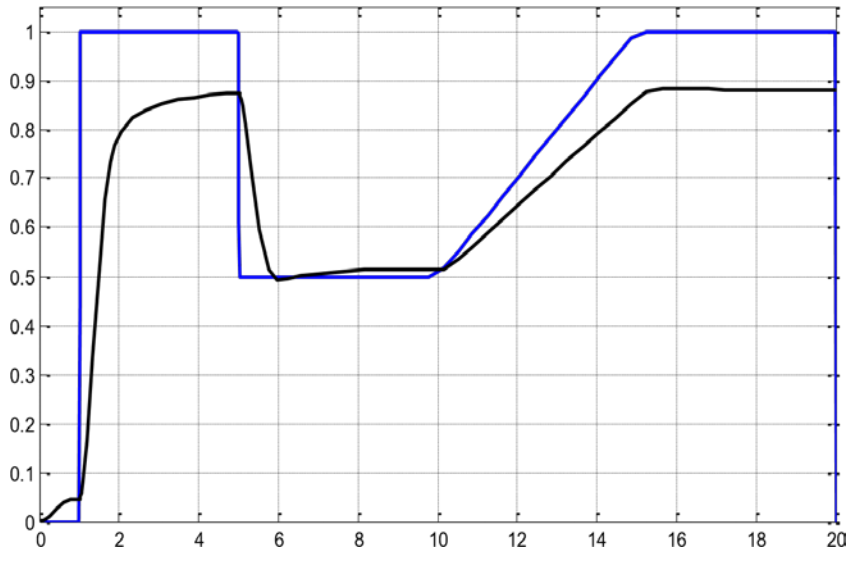


Fig. 12. Răspunsul sistemului neuro-fuzzy cu baza de reguli 7 x 5

Marele avantaj al RN, *capacitatea de învățare automată*, poate fi ilustrată prin varianta regulatorului cu doar 15 reguli, având baza de reguli cu dimensiunea 5 x 3. Răspunsul acestui regulator, prezentat în fig. 13, foarte asemănător cu răspunsurile regulatoarelor mai complicate, justifică alegerea sa pentru aplicații, având în vedere simplitatea și costul mai redus, la performanțe asemănătoare.

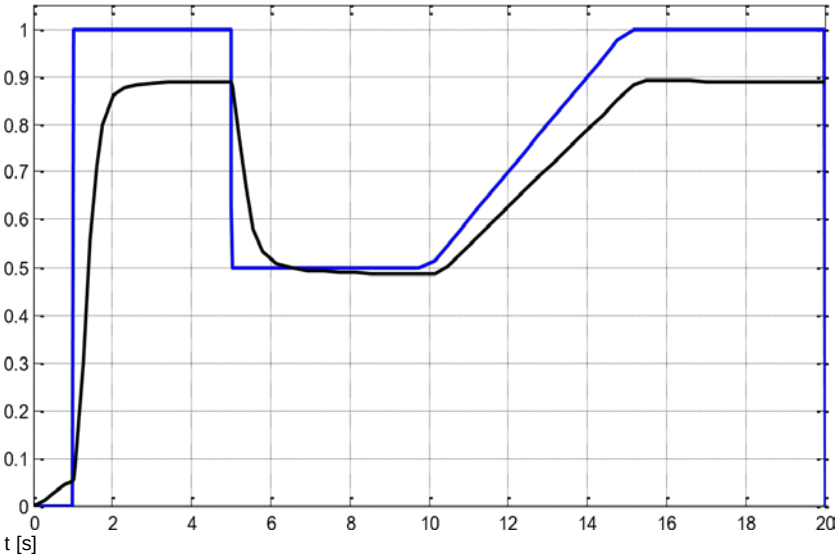


Fig. 13. Răspunsul sistemului neuro-fuzzy cu baza de reguli 5 x 3

Fereastra de dialog prin care se alege structura FIS asociată, în acest caz cu 15 reguli, precum și tipul mulțimilor fuzzy de la ieșire, în acest caz linear, este prezentată în figura următoare. Această fereastră de dialog apare prin acționarea butonului **Generate FIS**.

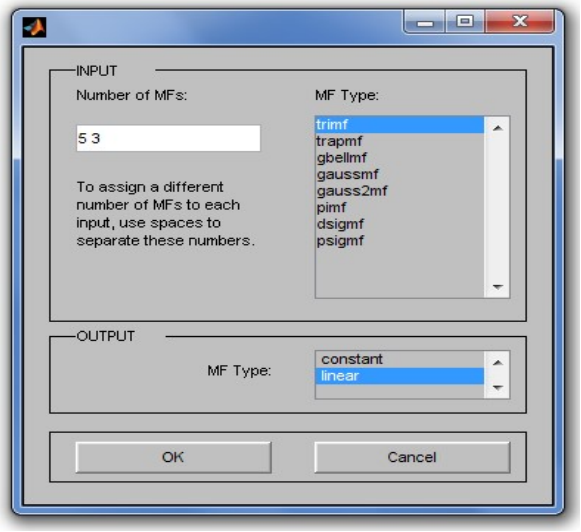


Fig. 14. Generarea FIS-ului asociat ANFIS

Dezavantajul principal al RN este acela că ele maschează cunoștințele referitoare la structura și ajustarea lor. Din valorile numerice ale TL variabilei de ieșire, ca de altfel din valorile parametrilor oricărui neuron, nu se poate trage absolut nici o concluzie cu privire la o eventuală strategie care să le justifice. Aceste valori sunt practic aleatoare, iar în cazul în care datele de testare sunt mult diferite de cele de antrenare întreaga aplicație poate eșua.

Desfășurarea lucrării

Această lucrare se constituie într-un îndrumător de proiect. Pornind de la metodologia prezentată studenții vor întocmi un proiect de sistem neuro-fuzzy, cu ajutorul tool-kitului

ANFIS. Tema proiectului poate fi aleasă din lista următoare, dar studenții interesați în alte teme sunt încurajați să le abordeze.

aprofundarea regulatorului PD neuro-fuzzy și încadrarea sa într-un sistem de reglare modelat în Simulink: reglarea turației unor motoare electrice, conducerea unui proces de frânare, conducerea automată a unui automobil, robotică, etc.

realizarea unui regulator PID neuro-fuzzy , optimizarea unor modele Simulink, etc.