

5.4.- MODUL DE LUCRU IN "COMMAND LINE"

Pentru a se deschide un FIS existent in F-L-T:
>> a=readfis('tipper.fis'), sau: ('f_log_1.fis').

MATLAB raspunde:

a =

```

    name: 'tipper'
    type: 'mamdani'
    andMethod: 'min'
    orMethod: 'max'
    defuzzMethod: 'centroid'
    impMethod: 'min'
    aggMethod: 'max'
    input: [1x2 struct]
    output: [1x1 struct]
    rule: [1x3 struct]

```

Etichetele din stanga listei reprezinta diferitele componente ale structurii MATLAB asociate cu "tipper.fis".

OBS.: "tipper.fis" este analog cu "F_log_1.fis" din cap. 5.3.2, variabilele avand alte nume: labour → service; efficiency → food; salary → tip, referindu-se la un "dineu la restaurant".

Se pot accesa componente ale structurii MATLAB, asociate cu "tipper.fis", cu comenzi de tipul:

```

>> a.type
ans =
    mamdani

```

Analog:

```

a.input
a.input(1), mf(1)

```

Funcția "getfis(a)" returneaza aproximativ aceleasi informatii ca si "a" (de mai sus):

```

>> getfis(a)
    Name    = tipper
    Type    = mamdani
    NumInputs = 2
    InLabels =
        service
        food
    NumOutputs = 1
    OutLabels =
        tip
    NumRules = 3
    AndMethod = min
    OrMethod = max
    ImpMethod = min
    AggMethod = max
    DefuzzMethod = centroid

```

ans =

```

    tipper

```

Pentru ca unele campuri de mai sus nu sunt parti ale structurii "a", nu se obtin informatii tastand "a.Inlabels", ci "getfis(a,'Inlabels')", sau – similar:

```

getfis(a, 'input',1)
getfis(a, 'output',1)
getfis(a, 'input',1,'mf',1).

```

Funcția "getfis" este modelata dupa functia Handle Graphics "get". Exista si "setfis", care permite schimbarea proprietatilor unui FIS; de ex. schimbarea numelui:

```

a=setfis(a,'name','the job');

```

sau, mai simplu: a.name='the job';

Apelarea GUI-Tools:

```

>> fuzzy(a) → va apare FIS-E;

```

```

>> mfedit(a) → MF-E;

```

```

>> ruleedit(a) → R-E;

```

```

>> ruleview(a) → R-V;

```

```

>> surfview(a) → S-V.

```

Dupa deschiderea unui GUI, altul se deschide mai comod cu meniul pull-down.

5.4.1.- VIZUALIZAREA FUNCTIILOR

Exista trei functii pentru vizualizarea FIS: plotfis, plotmf si gensurf.

```

>> plotfis(a) → apar trei careuri (de sus) din FIS-E (fig. 5.3.1).

```

Dupa inchiderea unei figuri MATLAB, sau fereastra GUI, functia "plotmf" ploteaza toate MF asociate unei variabile date:

```

>> plotmf(a,'input',1) → fig. 5.4.1.

```

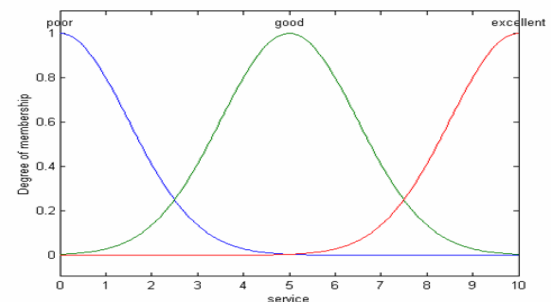


Fig. 5.4.1

```

>> plotmf(a,'output',1) → fig. 5.4.2.

```

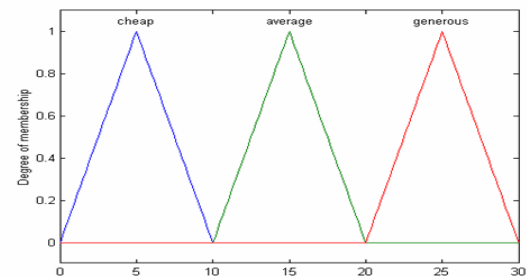


Fig. 5.4.2

Aceste ploturi vor apare in MF-E sau intr-o figura MATLAB, daca "plotmf" este apelata cand una, sau cealalta, este deschisa.

```

>> gensurf (a) → ploteaza oricare iesire, in functie de una sau doua intrari.

```

5.4.2- CONSTRUIREA UNUI FIS, PORNIND DE LA ZERO

Regulile se pot construi cu functia “addrule”. Fiecare variabila (de intrare, sau de iesire) si fiecare MF are un index. Regulile se construiesc cu propozitii de forma:

*if input1 is MF1 or input2 is MF3
then output1 is MF2 (weight = 0.5)*

Aceasta regula se transforma intr-o structura, pe baza urmatoarei logici: daca sistemul are m intrari si n iesiri, atunci primele m numere din structura regulii corespund intrarilor de la 1 la m . Numarul din coloana 1 este indexul pentru MF asociata cu intrarea 1; numarul din coloana 2 este indexul pentru MF asociata cu intrarea 2; s.a.m.d. Analog – urmatoarele n coloane, referitoare la iesiri. Coloana $m + n + 1$ reprezinta ponderea asociata cu aceasta regula (uzual = 1), iar coloana $m + n + 2$ reprezinta conectivul (AND = 1 si OR = 2). Structura regulii de mai sus are forma:

1 3 2 0.5 2.

O prima cale de construire a unui FIS, utilizand linii de comenzi in sintaxa MATLAB, este cu functia “newfis”.

A doua cale utilizeaza comenzile F-L-T:

```
a=newfis('tipper');
a=addmf(a,'input',1,'service',[0 10]);
a=addmf(a,'input',1,'poor','gaussmf',[1.5 0]);
a=addmf(a,'input',1,'good','gaussmf',[1.5 5]);
a=addmf(a,'input',1,'excellent','gaussmf',[1.5 10]);
a=addvar(a,'input','food',[0 10]);
a=addmf(a,'input',2,'rancid','trapmf',[-2 0 1 3]);
a=addmf(a,'input',2,'delicious','trapmf',[7 9 10 12]);
a=addvar(a,'output','tip',[0 30]);
a=addmf(a,'output',1,'cheap','trimf',[0 5 10]);
a=addmf(a,'output',1,'average','trimf',[10 15 20]);
```

```
a=addmf(a,'output',1,'generous','trimf',[20 25 30]);
ruleList=[...
1 1 1 1 2
2 0 2 1 1
3 2 3 1 2 ];
a=addrule(a,ruleList);
```

5.4.3- EVALUAREA FIS

De ex., evaluam “tipper” pentru intr. [1 2]:

```
>> a=readfis('tipper');
>> evalfis([1 2],a)
ans =
5.5586
```

Facand mai multe evaluari simultane, se realizeaza o crestere de viteza:

```
>> evalfis([3 5;2 7],a)
ans =
12.2184
7.7885
```

5.5.- INFERENTA FUZZY DE TIP SUGENO

Aceasta este – in multe privinte - similara inferentei Mamdani (de mai sus), primele doua parti ale ale procesului (fuzificarea intrarilor si aplicarea operatorului fuzzy) fiind identice. Principala deosebire este ca la inferenta Sugeno MF a iesirilor sunt numai liniare, sau constante.

O regula fuzzy pentru un *model fuzzy Sugeno de ordinul zero* are forma:

if x is A and y is B then z = k,

unde A si B sunt multimi fuzzy, iar k este o constanta. Deci toate MF a iesirilor sunt singleton-uri.

In figura 5.5.1 se arata prelucrarea uneia din cele trei reguli ale sistemului din cap. 5.2

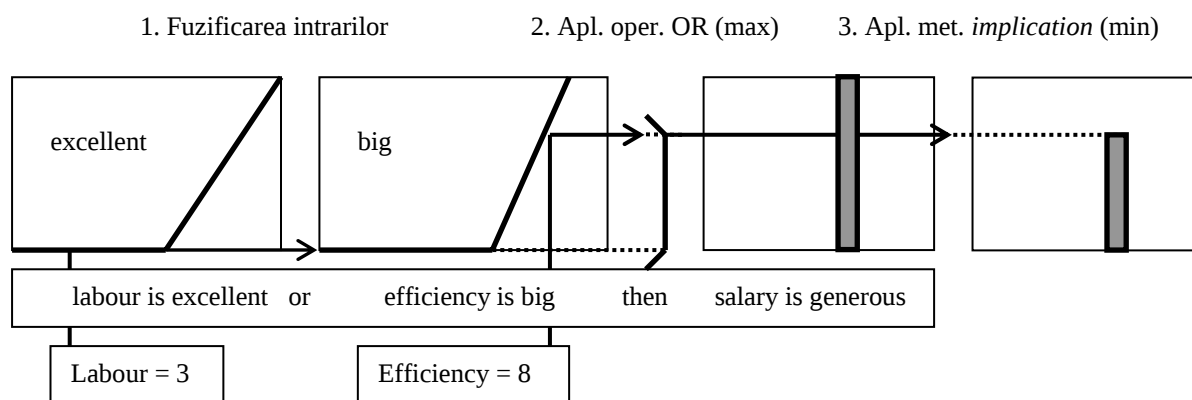


Fig. 5.5.1

Agregarea tuturor iesirilor se realizeaza suprapunand careul din dreapta figurii 5.5.1 cu careurile omoloage din celelalte doua reguli, rezultand fig. 5.5.2.

Defuzificarea se face – de regula – cu metoda *mediei ponderate*, adica se ia suma produselor dintre cele trei abscise cu ordonatele corespunzatoare si se imparte la suma

ordonatelor.

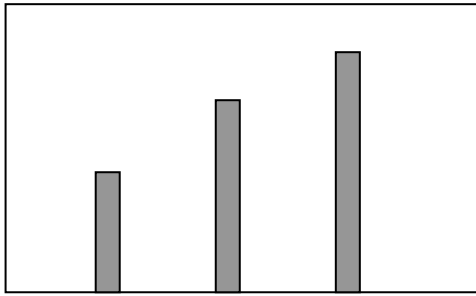


Fig. 5.5.2

Modelul fuzzy Sugeno de ordinul unu are forma:

if x is A and y is B then $z = p*x + q*y + r$, unde A si B sunt multimi fuzzy, iar p , q si r sunt constante. Acum in careul corespunzator iesirii din figura 5.5.1 apare un “singleton mobil”, lungimea lui avand o variatie liniara. Datorita dependentei liniare a fiecarei reguli de variabilele de intrare in sistem, metoda Sugeno este indicata ca un supervisor interpolator a mai multor controlere liniare aplicate diferitelor conditii de operare a unui sistem dinamic *neliniar*. De ex., comportarea unui avion se poate schimba brusc cu altitudinea si viteza. Controlerele liniare, desi sunt usor de implementat si se adapteaza oricaror conditii de zbor, trebuie reglate des pentru a se adapta cu starile variabile ale aeronavei. Un FIS Sugeno este cel mai potrivit in acest caz, pentru ca poate interpola cu usurinta amplificatorii liniari aparuti in spatiul intrarilor. Analog, un sistem Sugeno este indicat pentru modelarea sistemelor neliniare, prin interpolarea mai multor modele liniare.

5.5.1- UN EXEMPLU: “DOUA LINII”

Consideram un sistem cu o intrare si o iesire, existent in “sugeno1.fis”:

```
>> fismat=readfis('sugeno1');
```

```
>> getfis(fismat,'output',1)
```

```
Name = output
```

```
NumMFs = 2
```

```
MFLabels =
```

```
line1
```

```
line2
```

```
Range = [0 1]
```

Variabila de iesire are doua MF:

```
>> getfis(fismat,'output',1,'mf',1)
```

```
Name = line1
```

```
Type = linear
```

```
Params = [-1 -1]
```

```
>> getfis(fismat,'output',1,'mf',2)
```

```
Name = line2
```

```
Type = linear
```

Params = [1 -1]

Acestea sunt functii liniare de variabila de intrare. MF *line1* este definit de ecuatie:

$$\text{output}=(-1)*\text{input}+(-1),$$

iar MF *line2* este definit de ecuatie:

$$\text{output}=(1)*\text{input}+(-1).$$

MF ale intrarilor si regulile precizeaza care dintre functiile de iesire se vor activa si cand.

```
>> showrule(fismat)
```

```
ans =
```

```
1. If (input is low) then (output is line1) (1)
```

```
2. If (input is high) then (output is line2) (1)
```

Functia “plotmf” ne arata ca MF *low* se refera la valori ale intrarii < 0 , iar *high* – la valori > 0 .

```
>> subplot(211),plotmf(fismat,'input',1); → fig. 5.5.3-a.
```

Functia “gensurf” arata cum iesirea din sistemul fuzzy se comuta usor de la linia numita *line1* la *line2*:

```
>> subplot(212),gensurf(fismat) → fig. 5.5.3-b
```

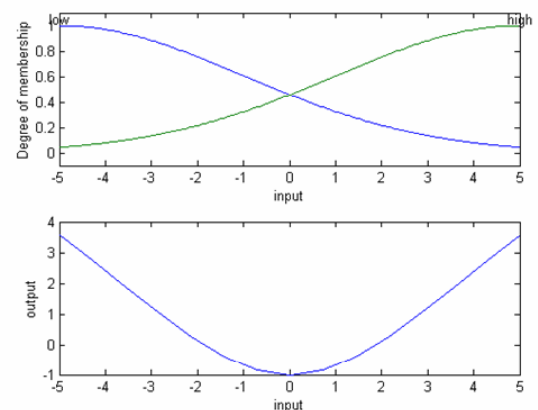


Fig. 5.5.3

Acest exemplu arata cum sistemul de tip Sugeno permite incorporarea sistemelor liniare in sisteme fuzzy. Extrapoland, se poate construi un sistem fuzzy care sa comute intre cateva controlere liniare si optimale, in timp ce sistemul neliniar se misca in spatiul sau.

5.6.- “anfis” si ANFIS EDITOR GUI

Pana acum am studiat structuri de FIS care transmit intrarile in sistem la intrarile in MF, intrarile in MF la reguli, regulile la iesirile din ele, aceste iesiri la iesirile din MF si iesirile din MF la o valoare unica a iesirii din sistem. Am lucrat numai cu MF tipice si alese cam arbitrar. Asadar, am aplicat inferenta fuzzy pentru modelarea sistemelor la care structura regulilor este predeterminata de interpretarea pe care am dat-o caracteristicilor variabilelor din model.

În continuare vom prezenta funcția “anfis” și ANFIS Editor GUI din F-L-T. Aceste instrumente aplică tehnicile de inferență fuzzy pentru modelarea datelor. Vom vedea cum, folosind aceste aplicații din F-L-T, parametrii MF se vor alege automat.

Considerăm că vrem să aplicăm inferență fuzzy unui sistem pentru care avem o mulțime de date de i/o pe care le utilizăm pentru modelare. Nu este necesar să avem o structură predeterminată a modelului. Sunt cazuri în care – pe baza datelor – nu se poate discerne ce MF trebuie alese. În locul unei alegeri a parametrilor pentru o MF arbitrară, acești parametri se pot alege astfel ca să adapteze MF datelor de i/o, utilizând *metoda de instruire neuro-adaptivă* (din “anfis”). Aceste tehnici furnizează, pentru modelarea fuzzy, o metodă de instruire a informațiilor privind o mulțime de date, cu scopul calculării parametrilor MF care permit cel mai bine FIS-ului asociat să urmărească datele de i/o. Această metodă de instruire acționează similar cu RNA. Funcția F-L-T care realizează această ajustare a parametrilor MF se numește “anfis” și se poate accesa prin linie de comandă, sau prin ANFIS Editor GUI. ANFIS este acronimul de la *adaptive neuro-fuzzy inference system*. Plecând de la un set de date de i/o, “anfis” construiește un FIS la care parametrii MF sunt ajustați utilizând un algoritm “backprop”, singur sau combinat cu o metodă a resturilor patratică. Aceasta îi permite sistemului fuzzy să se instruiască pe baza datelor pe care le modelează. Pentru interpretarea spațiului de i/o se utilizează o structură de tip rețea – similară cu o RNA – ce prelucrează intrările în sistem spre intrările în MF și parametrii asociați, apoi ieșirile din MF și parametrii asociați spre ieșirile din sistem.

Parametrii asociați cu MF se vor modifica în timpul procesului de instruire. Calculul (sau modificarea) acestor parametri se face cu un vector gradient, care indică măsura în care FIS modelează datele de i/o pentru un set dat de parametri. După obținerea vectorului gradient, pentru ajustarea parametrilor se poate aplica orice rutină de optimizare, în vederea reducerii erorii (definite ca suma patratelor diferențelor dintre ieșirile actuale și cele dorite).

Modelarea cu “anfis” este similară multor tehnici de identificare a sistemelor. Mai întâi se realizează o structură parametrizată a modelului (relaționând intrările la MF, pe acestea la reguli, apoi la ieșiri, etc.). În continuare se colectează date de i/o, într-o formă adecvată pentru ca “anfis” să le utilizeze pentru antrenare. Apoi “anfis” antrenează modelul FIS în vederea ameliorării datelor ce-i sunt prezentate, modificând parametrii MF corespunzător cu criteriul de eroare folosit.

Acest tip de modelare este eficient dacă datele de antrenare prezentate “anfis”-ului pentru estimarea parametrilor MF sunt integral reprezentative pentru caracteristicile datelor pe care FIS-ul antrenat le va modela. Acest lucru nu se întâmplă frecvent, uneori datele fiind afectate de zgomot de măsurare. Acesta este momentul în care *validarea modelului* intra în joc.

5.6.1- VALIDAREA MODELULUI PE BAZA DATELOR PENTRU VERIFICARE ȘI TESTARE

Validarea modelului este procesul în care vectorii de intrare din setul de date de i/o, care nu s-au folosit la antrenarea FIS, sunt prezentați modelului FIS antrenat, pentru a vedea cât de bine modelul FIS estimează valorile corespunzătoare ieșirii din setul de date. Aceasta se realizează cu *testing data set* din ANFIS Ed. GUI.

Pentru validarea modelului, “anfis” mai are un alt tip de set de date, numit *checking data set*, utilizat pentru a vedea potențialul modelului de a se potrivi cu datele. Când *checking data* este prezentat “anfis”-ului, analog cu datele de antrenare, modelul FIS se alege astfel ca să aibă parametrii asociați corespunzători erorii minime a modelului datelor de verificare.

O chestiune ce apare la validarea modelelor construite folosind tehnici adaptive este selectarea unui set de date care să fie reprezentative pentru datele pe care modelul este antrenat să le modeleze și – în plus – să fie suficient de diferite de setul de date de antrenare, astfel ca validarea să nu devină trivială. Dacă s-a colectat o mare cantitate de date, sunt speranțe ca acestea să continue toate trăsăturile reprezentative necesare, astfel ca

selectarea unui set de date, in vederea verificarii sau testarii, este usoara. Totusi, daca se apreciaza ca zgomotul de masurare este prezent, este posibil ca setul de date de antrenare sa nu includa toate trasaturile reprezentative.

Ideea de baza a utilizarii unui set de date de verificare pentru validarea modelului este aceea ca dupa un anumit moment al antrenarii, modelul incepe sa se potriveasca cu setul de date de antrenare. In principiu, eroarea modelului pentru setul de date de verificare tinde sa descreasca pana cand antrenarea depaseste momentul mentionat mai sus, pentru ca apoi sa creasca brusc.

5.6.2- LIMITARI ALE ANFIS

ANFIS este mai complex ca FIS, dar nu este valabil pentru toate variantele FIS, el suportand numai sisteme tip Sugeno, care trebuie sa fie:

- de ordinul 1 sau 0;
- cu o iesire, obtinuta prin defuzificarea mediei ponderate (MF a iesirii – liniara sau const.);
- cu ponderea 1 pentru fiecare regula.

Daca structura FIS nu indeplineste aceste constrangeri, se da eroare.

Mai mult, ANFIS nu accepta customizari, adica nu iti poti crea propriile MF si functii de defuzificare, ci se vor utiliza cele din dotare.

5.6.3- ANFIS EDITOR GUI

Se tast.: >>anfisedit; apare:

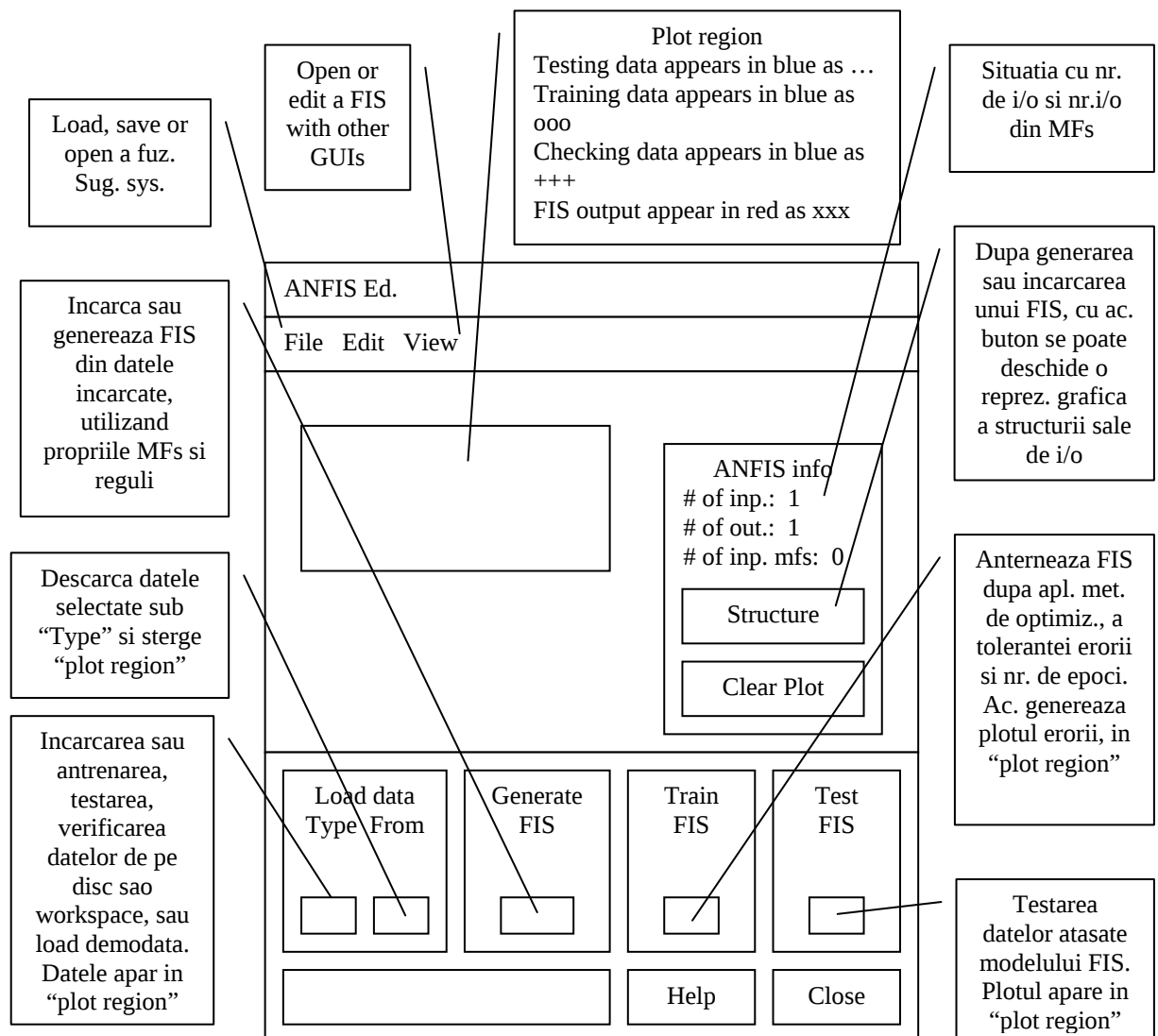


Fig. 5.6.1

Cu acest GUI se poate:

- incarca date (de antrenare, testare sau verificare), selectand in "Load data" butonul radio corespunzator, apoi "Load data...";

datele incarcate sunt plotate in zona "Plot".

- genera sau incarca un model FIS initial, cu butonul "Generate FIS";
- vizualiza structura unui model FIS, dupa ce

un model FIS initial a fost generat sau incarcat, cu butonul “Structure”;

- alege metoda de optimizare parametrica a modelului FIS: back propagation, sau o combinatie a lui cu resturi patratic (metoda hibrida).

- alege numarul epocilor de antrenare si a tolerantei erorii de antrenare;

- antrena modelul FIS → butonul “Train Now”; antrenarea ajusteaza parametrii MFs si ploteaza eroarea;

- vizualiza iesirea modelului FIS in functie de iesirea din datele de antrenare (sau verificare, sau testare) → butonul “Test Now”. Aceasta functie ploteaza datele de testare atasate iesirii din FIS, in zona “Plot”.

De asemenea se poate utiliza bara meniu a lui ANFIS Editor GUI pentru a incarca o initializare a antrenarii FIS, a salva FIS-ul antrenat, a deschide un nou sistem Sugeno sau sa deschidem orice alt GUIs pentru a interpreta modelul FIS antrenat.

TESTARE SI ANTRENARE

Pentru a incepe antrenarea unui FIS, utilizand “anfis” sau ANFIS Ed. GUI, mai intai trebuie sa avem un set de date de antrenare care sa contina perechile de date de i/o necesare pentru modelarea sistemului tinta.

Alteori trebuie sa avem un set de date de testare cu care sa verificam capacitatea de generalizare a FIS-ului rezultat si / sau un set de date de verificare, utilizate la ajustarea modelului in timpul antrenarii. Utilizarea setului de date de testare si a setului de date de verificare pentru validarea modelului, s-a prezentat mai sus. Asa cum s-a aratat, potrivirea se realizeaza prin testarea FIS-ului antrenat cu datele de antrenare, prin comparare cu datele de verificare si alegerea parametrilor MFs astfel ca sa aiba eroarea de verificare minima (aceasta eroare indicand potrivirea modelului). Pentru a realiza acest obiectiv, se vor examina atent graficele erorii de antrenare. Uzual, seturile de date pentru antrenare si verificare se colecteaza pe baza observarii sistemului tinta si sunt stocate in fisiere separate.

NOTA: Orice set de date ce se incarca in ANFIS Editor GUI (sau in linia de comanda “anfis”) trebuie sa fie o matrice cu datele de intrare aranjate ca vectori, cu exceptia ultimei coloane, in care se pun datele de iesire.

5.7.- CLUSTERIZAREA FUZZY

Clusterizarea datelor numerice este baza multor clasificari si algoritmi de modelare ale sistemelor. Scopul clusterizarii este de a identifica grupurile naturale dintr-o multime de date. F-L-T are instrumente care permit gasirea clusterilor in datele de i/o. Cu informatiile despre clusteri se poate genera un FIS de tip Sugeno, care modeleaza optim comportarea datelor folosind un set minim de reguli. Regulile fac ele inele partitia, pe baza calitatilor fuzzy asociate fiecarui cluster de date. Acest tip de generare a unui FIS se poate realiza automat utilizand in linia de comanda functia *genfis2*

5.7.1- CLUSTERIZAREA FUZZY A MEDIILOR C

Aceasta este o tehnica de clusterizare a datelor la care fiecare punct de date apartine unui cluster, cu un anumit grad de apartenenta.

In F-L-T exista functia “fcm” care incepe cu alegerea centrelor clusterilor (in locatia medie a fiecarui cluster). Aceasta alegere este cel mai adesea inexacta; ca urmare “fcm” atribuie fiecarui punct de date un grad de apartenenta pentru fiecare cluster. Corectand iterativ centrele clusterilor si gradele de apartenenta, “fcm” deplaseaza centrele clusterilor spre locatia lor corecta. Aceasta iteratie se bazeaza pe minimizarea unei functii obiectiv ce reprezinta distanta de la un punct de date la centrul clusterului, ponderata cu gradul de apartenenta respectiv.

Iesirea din “fcm” este o lista a centrelor clusterilor si a gradelor de apartenenta pentru fiecare punct de date. Cu aceste informatii se poate construi un FIS creand MF pentru calitatile fuzzy ale fiecarui cluster.

Exemplu: CLUSTERI IN PLAN

Incarcam un set de date:

```
>> load fcmdata.dat  
>> plot (fcmdata(:,1),fcmdata(:,2),'o')
```

Apare figura 5.7.1.

Acum apelam functia “fcm” si ii cerem sa gaseasca doi clusteri in acest set de date,

```
>> [center,U,obj_fcn]=fcm(fcmdata,2);  
Iteration count = 1, obj. fcn = 8.933363  
Iteration count = 2, obj. fcn = 6.910409  
Iteration count = 3, obj. fcn = 5.431071  
Iteration count = 4, obj. fcn = 4.082928  
Iteration count = 5, obj. fcn = 3.830947  
Iteration count = 6, obj. fcn = 3.802926
```

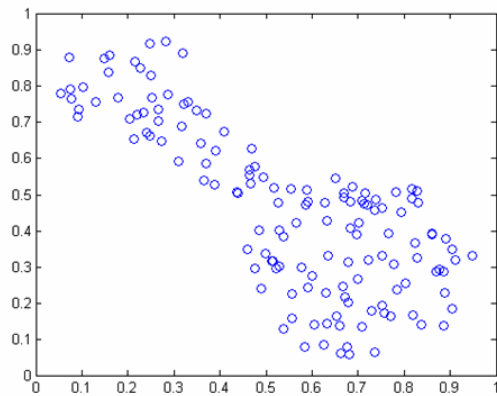


Fig. 5.7.1

Iteration count = 7, obj. fcn = 3.798428
 Iteration count = 8, obj. fcn = 3.797611
 Iteration count = 9, obj. fcn = 3.797462
 Iteration count = 10, obj. fcn = 3.797435
 Iteration count = 11, obj. fcn = 3.797430,

pana cand functia obiectiv nu mai descreste semnificativ.

Variabila “center” contine coordonatele centrelor a doi clusteri, “U” contine gradele de apartenenta pentru fiecare dintre punctele de date, iar “obj_fcn” contine evolutia functiei obiectiv.

Functia “fcm” este o bucla iterativa ce contine subrutinele: “initfcm” (care initializeaza problema), “distfcm” (calculeaza distantele), “stepfcm” (care face o iteratie completa).

Plotul functiei obiectiv arata progresul clusterizarii: `>> plot(obj_fcn)` → Fig. 5.7.2.

In programul de mai jos “fcm” clasifica doi clusteri:

```
[center,U,obj_fcn]=fcm(fcmdata,2);
maxU=max(U);
index1=find(U(1,:)==maxU);
index2=find(U(2,:)==maxU);
line(fcmdata(index1,1),fcmdata(index1,2),'linestyle','none','marker','o','color','g');
line(fcmdata(index2,1),fcmdata(index2,2),'linestyle','none','marker','x','color','r');
```

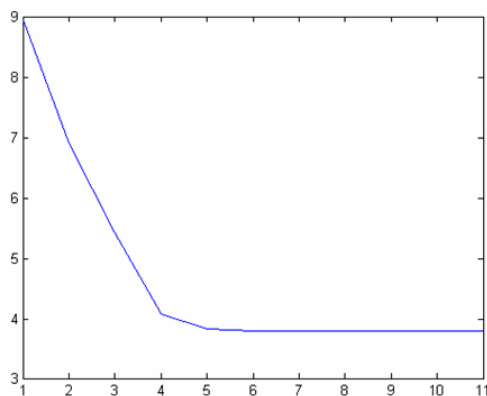


Fig. 5.7.2

```
hold on
plot(center(1,1),center(1,2),'ko','markersize',15,'LineWidth',2)
plot(center(2,1),center(2,2),'kx','markersize',15,'LineWidth',2);
```

In figura 5.7.3 centrele clusterilor se indica cu caractere **bold**.

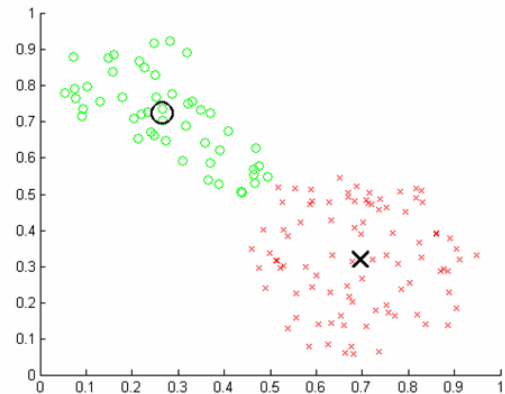


Fig. 5.7.3

5.8- LUCRUL CU SIMULINK

SIMULINK este acronimul pentru cuvintele *simulation* si *link* (legatura). Acest termen face trimitere la “simulare”, adica reproducerea artificiala a ce se intampla, sau s-ar putea intampla, in lumea reala. SIMULINK este un toolbox in MATLAB 6.5.

5.8.1- CART & POLE

Consideram problema: *DINAMICA CULISEI CU UN PENDUL INVERSAT (CART & POLE Simulation)*, din fig. 5.8.1 (Sa se proiecteze un controler care sa echilibreze un pendul montat – inversat – cu o articulatie, de o culisa si sa mentina culisa la o anumita pozitie. Pentru realizarea echilibrului se actioneaza cu o forta orizontala c)

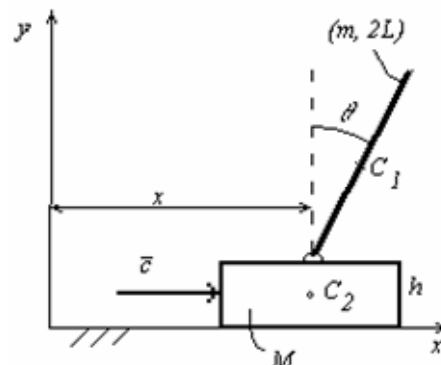


Fig. 5.8.1

Cinematica acestui sistem:

$$\begin{aligned}
\bar{r}_{C_1} &= (x + L \cdot \sin \theta) \bar{i} + (h + L \cdot \cos \theta) \bar{j}; \\
\bar{v}_{C_1} &= (\dot{x} + L \cdot \dot{\theta} \cdot \cos \theta) \bar{i} + (-L \cdot \dot{\theta} \cdot \sin \theta) \bar{j}; \\
\bar{a}_{C_1} &= (\ddot{x} + L \cdot \ddot{\theta} \cdot \cos \theta - L \cdot \dot{\theta}^2 \cdot \sin \theta) \bar{i} - \\
&\quad - L(\ddot{\theta} \cdot \sin \theta + \dot{\theta}^2 \cdot \cos \theta) \bar{j}.
\end{aligned}
\tag{5.8.1}$$

Dinamica o studiem cu metoda lui d'Alembert (fig. 5.8.2):

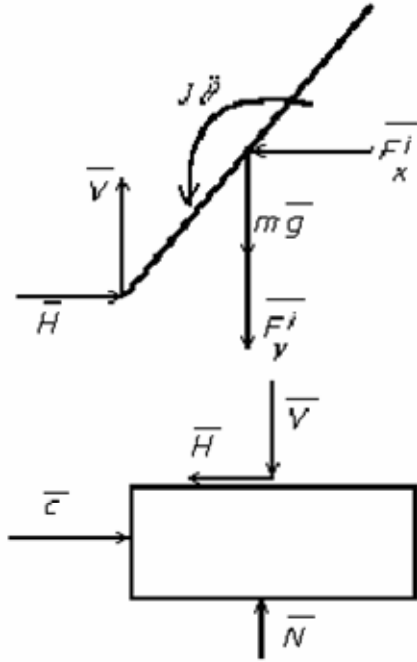


Fig. 5.8.2

$$\begin{aligned}
H &= m \cdot \ddot{x} + mL(\ddot{\theta} \cdot \cos \theta - \dot{\theta}^2 \sin \theta) \\
V &= mg - mL(\ddot{\theta} \cdot \sin \theta + \dot{\theta}^2 \cos \theta) \\
\frac{mL^2}{3} \ddot{\theta} + HL \cdot \cos \theta - VL \cdot \sin \theta &= 0 \\
M \ddot{x} &= c - H
\end{aligned}
\tag{5.8.2}$$

Rezolvam acest sistem de ecuatii:

$$\begin{aligned}
\frac{mL}{3} \ddot{\theta} + m \cdot \ddot{x} \cdot \cos \theta + mL \ddot{\theta} \cdot \cos^2 \theta - \\
- mL \dot{\theta}^2 \sin \theta \cdot \cos \theta - mg \sin \theta + mL \ddot{\theta} \cdot \sin^2 \theta + \\
+ mL \dot{\theta}^2 \cos \theta \cdot \sin \theta &= 0 \\
M \ddot{x} &= c - m \cdot \ddot{x} - mL(\ddot{\theta} \cdot \cos \theta - \dot{\theta}^2 \sin \theta)
\end{aligned}
\tag{5.8.3}$$

Deducem:

$$\begin{cases} \frac{4}{3} L \ddot{\theta} + \ddot{x} \cos \theta - g \sin \theta = 0 \\ (M + m) \ddot{x} + mL(\ddot{\theta} \cdot \cos \theta - \dot{\theta}^2 \sin \theta) - c = 0 \end{cases}
\tag{5.8.4}$$

$$\begin{cases} \ddot{x} = g \cdot \tan \theta - \frac{4L}{3 \cdot \cos \theta} \ddot{\theta} \\ M g \cdot \tan \theta - M \frac{4L}{3 \cdot \cos \theta} \ddot{\theta} + mL(\ddot{\theta} \cdot \cos \theta - \\ - \dot{\theta}^2 \sin \theta) - c = 0 \end{cases}$$

(5.8.5)

in care: $M = M + m$. (5.8.6)

Ecuatia diferentiala a lui θ :

$$\begin{aligned}
L \cdot \left(\frac{4M}{3 \cdot \cos \theta} - m \cdot \cos \theta \right) \cdot \ddot{\theta} &= M g \cdot \tan \theta - \\
- mL \cdot \dot{\theta}^2 \cdot \sin \theta - c; \\
L \cdot (1.33 \cdot M - m \cdot \cos^2 \theta) \cdot \ddot{\theta} &= M g \cdot \sin \theta - \\
- mL \cdot \dot{\theta}^2 \cdot \sin \theta \cdot \cos \theta - c \cdot \cos \theta; \\
\ddot{\theta} &= [g \cdot \sin \theta - \cos \theta \cdot (c + mL \cdot \dot{\theta}^2 \cdot \sin \theta) / \\
/ M] / [L \cdot (4/3 - m \cdot \cos^2 \theta / M)].
\end{aligned}
\tag{5.8.7}$$

Schema in SIMULINK pentru aceasta problema exista in MATLAB: >>slcp – En → apare schema din figura 5.8.3. Clic dreapta pe blocul “Cart & Pole Dynamics”- Look under mask → apare figura 5.8.4. Pentru implementarea in SIMULINK a schemelor din figurile 5.8.3 si 5.8.4 (comasate), plecand de la zero, procedam astfel:

La promptul MATLAB tastam: >>simulink – En (sau clic pe butonul cu rosu din bara de sus)– Files – New – Model – Se deschide fereastra “Untitled” - In fereastra din stanga “...Library...” clic pe “Simulink”, apoi (mai jos) pe “Sources” - Din semi-fereastra din dreapta se trage blocul “Signal Generator” {Produce una dintre urmatoarele forme de unda: sinus, patrata, dinte de ferastau, zgomot aleator cu distr. uniforma} – Dublu clic pe bloc - Se deschide o caseta: in “Wave form” se alege “square”; la “Amplitudine” → 1.5; “Frequency” (= **pulsatia**) → 0.628 = $\pi/5$ rad/s. Schimbam eticheta in “Target Position”. Scrierea unei etichete: Dublu clic stanga la pozitia dorita - Se tasteaza textul – Clic in alta parte. Se poate muta in alta pozitie, cu butonul stang. Stergerea unei etichete: Clic in ea – Se sterge textul - Clic in alta parte.

Se mai trage blocul (din dreapta figurii 5.8.3) “Scope” (osciloscopul) si cele doua blocuri “Constant” (din partea stanga a figurii 5.8.3).

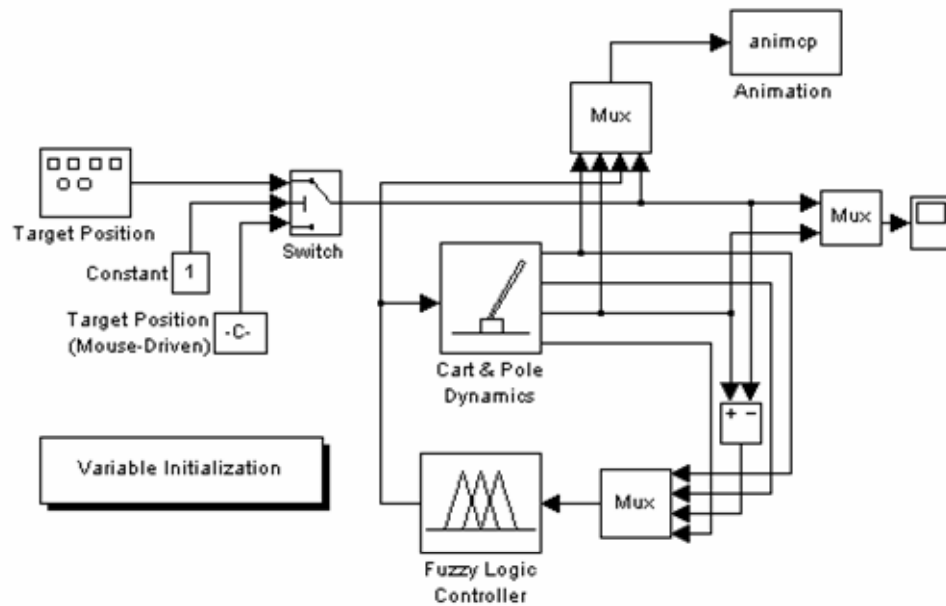


Fig. 5.8.3

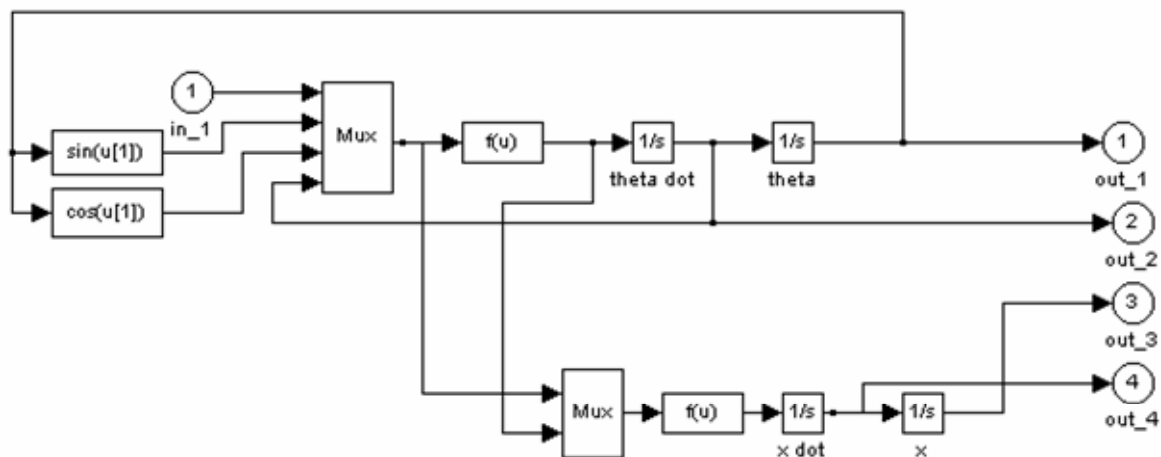


Fig. 5.8.4

Dublu clic pe blocul “Constant” – In caseta ce se deschide se tasteaza: “Constant value” → 0.1538, pentru al doilea bloc, caruia i se schimba si eticheta in “Target ...”.Clic (in stanga) pe “Nonlinear” – Se trage “Switch” (comutator) {Iesirea este egala cu prima intrare cat timp semnalul de comanda (a doua intrare) este $\geq$ decat o valoare de prag; in caz contrar, iesirea este = cu a treia intrare}. Dublu clic pe bloc – “Threshold” (valoarea de prag) → 0. Clic (in stanga) pe “Signals...” – Se trag succesiv cinci blocuri “Mux” {Are mai multe intrari, dar o singura iesire; el realizeaza combinarea a n semnale scalare intr-un singur canal de iesire} - Dublu clic pe bloc – “Nr. of inputs” → 4 (de ex.); “Display option” → none.

De exemplu in figura 5.8.4 un bloc “Mux” are patru intrari: $u(1)=c$; $u(2)=\sin\theta$;

$$u(3)=\cos\theta; \quad u(4)=\dot{\theta}; \quad (5.8.8)$$

la blocul “Mux” de jos mai apare o a cincea intrare: $u(5)=\ddot{\theta}$. (5.8.9)

Rotirea blocurilor: selectare – clic dreapta – Format ... , sau cu butonul FORMAT ...

Se mai trag blocurile “Inport” {Defineste un port de intrare [utilizat in operatia *group*, sau pentru a crea o *S-function* (explicate mai jos)]. Se numereaza cu numere naturale, incepand cu 1 (care este cel mai de sus)} si “Outport” {Analog cu precedentul, dar pentru iesire}.

Dublu clic pe blocul “in_1” din fig. 5.8.4 – In campul “Port number” se tasteaza: 1; in “Port

dimens. (-1 for dynamically sized)": -1; in "Sample time": -1; in rest: auto.

Clic (in stanga) pe "Functions..." – Se trag patru blocuri "Fcn", notate in fig. 5.8.4 cu "cos...", "sin..." si "f(u)". {"Fcn" este cel mai versatil bloc. El permite implementarea oricarei functii de i/o exprimata printr-o relatie matematica. Intrarea se indica prin litera "u". De exemplu, functia:

$$y = \begin{cases} 5\cos(u) & \text{if } u > 3 \\ \sin(u*r) & \text{if } 2 < u \leq 3 \\ u^2 & \text{if } u \leq 2 \end{cases}$$

unde r este o constanta, are forma (in blocul "Fcn"): $5*\cos(u)*(u>3)+\sin(r*u)*(u>2\&\&u\leq 3)+u^2*(u\leq 2)$.

"Fcn" poate lucra si cu mai multe semnale la intrare, notate: $u(1)$, . . . $u(n)$. Iesirea din "Fcn" este intotdeauna o valoare scalara} - Dublu clic pe blocul "Fcn" – In campul "Expression" se tasteaza functia dorita. De exemplu, blocul "f(u)" din figura 5.8.4 (centru sus), avand intrarile (5.8.8) si iesirea $\ddot{\theta}$, implementeaza functia (5.8.7):

$$\frac{(g*u(2)-u(3)*(u(1)+m*l*u(4)*u(4)*u(2))/(mc+m))/(1*(4/3-m*u(3)*u(3))/(mc+m)),}{(5.8.10)}$$

in care: $mc = M$; $l = L$, (5.8.11)

iar blocul "f(u)" din figura 5.8.4 (centru jos), avand intrarile (5.8.8), (5.8.9) si iesirea $\ddot{x}$, implementeaza prima fun. din (5.8.5):

$$g*u(2)/u(3)-4*l*u(5)/(3*u(3)). \quad (5.8.12)$$

Se mai trage in figura 5.8.3 (dreapta sus) un bloc "S-function" {Acesta permite includerea unei *S-function* intr-o schema Simulink. O *S-function* este un fisier ASCII care implementeaza o functie generica prin: - o schema Simulink; -o structura tip MATLAB; -o procedura C sau Fortran (Mex-file)} – Dublu clic pe bloc – In campul "S-function name" se tasteaza: animcp, iar in "S-function param.": Clic (in stanga) pe "Math"- Din semi-fereastră din dreapta se trage blocul "Sum"- Se deschide o caseta: in "Icon shape" se alege "rectangular", iar in "List of signs" tastam "+-" - OK – Se roteste blocul.

Clic (in stanga) pe "Continuous"- Se trag succesiv patru blocuri "Integrator".

Clic (in stanga), mai jos de "Simul.", pe "Fuzzy Logic Toolbox" (sau se tasteaza: >>fuzblock – En) – Se trage blocul "Fuzzy Logic Controller" (FLC) {In stanga mai este

blocul "FLC with R-V" care este o extensie a lui "FLC", permitand vizualizarea prelucrării regulilor in timpul simulării. Daca se trage "FLC with R-V" si se face clic dreapta pe el – Look under mask – Apare figura 5.8.5, in care

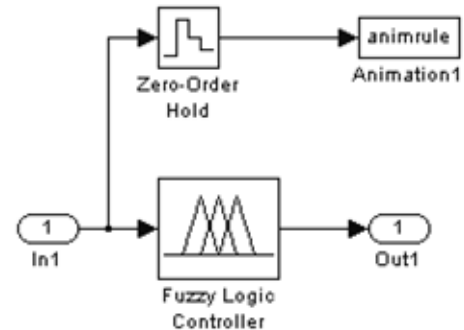


Fig. 5.8.5

blocul "Zero-Order Hold" este din grupul (din stanga) "Discrete". Acest bloc implementeaza si retine o functie esantionata, avand ca parametru *perioada de esantionare*. El discretizeaza semnalele: esantioneaza semnalul da la intrare la un anumit moment si retine aceasta valoare la iesire, pana la urmatorul moment.} – Pentru a initia FLC (cu sau fara R-V): dublu clic pe bloc – In campul "FIS File or Structure" se tasteaza: fismatrix . {Blocul FLC genereaza automat o reprezentare cu o diagrama bloc pentru FIS (abilitate numita *Fuzzy Wizard*), utilizand blocuri existente in Simulink. *Fuzzy Wizard* nu poate manipula FIS cu MF custom, sau cu functiile AND, OR, IMP si AGG in afara celor de mai jos: -orMethod: max; -andMethod: min, prod; -impMethod: min, prod; -aggMethod: max. In aceste cazuri, blocul FLC utilizeaza S-function "sffis", pentru a simula FIS.} – Clic dreapta pe blocul FLC– Look under mask – Apare figura 5.8.6.



Fig. 5.8.6

Daca in figura 5.8.3 facem clic dreapta pe blocul FLC – Look under mask – Apare o figura analoaga cu fig. 5.8.6, dar in centru scrie FIS, iar dedesubt: FIS Wizard - Clic dreapta pe blocul FIS Wizard – Apare o fereastră mai mare...

Daca blocurile nu sunt aliniate, se trag (cu butonul din stanga) in pozitia corecta. Pentru trasarea conexiunilor intre blocuri, se procedeaza (de exemplu) astfel: Se pozitioneaza mausul la iesirea (>) din “f(u)” si se apasa butonul stg: promptul devine o cruce; mentind butonul apasat se misca mausul spre dreapta, pana la atingerea intrarii (>) in primul “Integrator”; daca operatia s-a facut corect, parantezele unghiulare dispar si dreapta se transforma in sageata - Analog se leaga cele doua integratoare - Se plaseaza promptul la mijlocul unei linii si se apasa tasta Ctrl: promptul devine o cruce; se misca mausul in jos si apoi la stanga (astfel s-a introdus un punct de legatura) - Cu tasta Ctrl apasata se traseaza o orizontala ce depaseste putin blocul “Mux” si apoi o verticala in sus; se selecteaza ultima linie si, cu clic stanga pe capatul ei se traseaza o orizontala la “Mux” - Daca dreptele nu perpendiculare, se face clic stanga in coltul ce trebuie modificat - Promptul devine un cerc - Se misca mausul pana ce coltul devine drept - Analog se conecteaza celelalte blocuri.

5.8.2- OPERATIUNEA “GROUP” (grupare)
Claritatea unei scheme bloc se deterioreaza odata cu cresterea numarului de blocuri. O schema complexa se poate descompune in subscheme mai simple, ceea ce este benefic pentru analiza si depanare (fiecare componenta se testeaza separat, apoi se testeaza intreaga schema).

Nu exista un criteriu unic care sa dea cea mai buna descomp. Cea mai utilizata este descrierea pe doua nivele, dar se poate merge si mai adanc.

Un numar de blocuri elementare se poate grupa intr-un super-bloc numit *group*, ce este similar cu blocurile din Simulink. O schema obtinuta prin gruparea blocurilor poate fi ulterior grupata intr-un super-grup, s.a.m.d.

Gruparea se realizeaza cu metoda numita *selection via bounding box* (selectare cu un careu limita): Clic intr-un punct din schema (unde nu este vre-un bloc sau o linie); butonul se tine apasat - Promptul devine o cruce neagra, cu o cruce alba in interior - Miscand mausul, apare un careu punctat - La terminarea apasarii, toate blocurile si liniile incluse (chiar partial) in careu sunt selectate -

CTRL+G (sau: Options – Group) – Apare un bloc ca in figura 5.8.7, numit ca in eticheta;

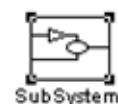


Fig. 5.8.7

eticheta se va modifica, pt. evitarea confuziei. Daca pe acest bloc se face dublu clic, se deschide o fereastră in care apare schema detaliata a grupului, la care intrarea este un *inport*, iar iesirea – un *outport*. Daca grupul are mai multe i/o, fiecare port se eticheteaza cu un numar, incepand cu 1. Simulink numeroteaza automat porturile, numarul 1 fiind cel mai de sus; aceasta ordine se poate modifica ulterior.

Daca porturile blocului “group” nu sunt specificate, Simulink le insereaza automat. De aceea putem sa nu tinem cont, punand porturile de i/o (utilizand blocurile din “Signals...”) inainte de operatia *group*.

CTRL+U (sau: Options – Ungroup) este operatia inversa.

5.8.3- UTILIZAREA MASTII (MASK)

Blocurile realizate cu optiunea “group” au acelasi icon, deosebindu-se numai prin nume si nu au “help”. Acest dezavantaj se poate surmonta mascand grupul: Se selecteaza grupul – Options – Mask – Se deschide o fereastră de dialog, in care sunt campurile:

- *New block type* → numele blocului.
- *Dialog strings separated by /* → etichetele parametrilor blocului (prima este titlul blocului).
- *Initialization commands* → se definesc comenzile de initializare si corespondenta intre parametrii ferestrei si ai variabilelor blocului. Stringurile ferestrei se acceseaza cu notatia: @1, @2, s.a.m.d. Aceste variabile sunt locale, neproducand nici-un efect exterior. Acest camp suporta orice expresie MATLAB.
- *Drawing commands* → se deseneaza un icon adecvat. Admite comanda *plot* (similara) cu cea din MATLAB.
- *Help string* → text ce apare la clic pe butonul “Help”.

La dublu clic pe blocul mascat se deschide o fereastră de dialog; este posibil ca – in locul acestei deschidei - sa se apeleze o functie

MATLAB, dacă în *Dialog strings...* se notează o funcție MATLAB inclusă în comanda “eval”.

De exemplu: Eval(‘load data_sat’) → la dublu

clic pe bloc, se va încărca fișierul “data_sat.mat”.

Iconul blocului se poate desena utilizând funcția MATLAB “iconedit”, ce permite desenarea cu mausul.