Fuzzy szabályok mért adatokból – BIR labor mérési utasítása

Fuzzy szabályok tanulása mért adatokból intelligens tér környezetben.

A probléma rövid leírása

Mérés során kísérletezünk egy fuzzy mérő/szabályozó rendszernek a mért adatok alapján történő kialakításával. Ilyen problémával találkozhatunk ott, ahol a vizsgált rendszer túlságosan bonyolult (pl. emberek lakta ambiens intelligens tér) ahhoz, hogy esélyünk lehessen egy előzetes analitikus modell létrehozására és a fuzzy rendszer modell alapú megtervezésére.

Valós adatokból épített fuzzy rendszernél először „felvételt kell készíteni” a szabályozás mikéntjéről. Pontosabban arról, hogy bizonyos beavatkozások milyen környezeti paraméterek mellett válnak kívánatossá és kerülnek végrehajtásra az intelligens tér lakói által. Az így megmért adatokhoz „meg kell sejteni” a legjobban illeszkedő fuzzy tagsági függvényeket, majd a tagsági függvények és a konkrét mért jelszekvenciák birtokában meghatározni a fuzzy rendszer fuzzy szabálybázisát, amely a mért adatokban rejlő információnak tömör és általánosított modellje.

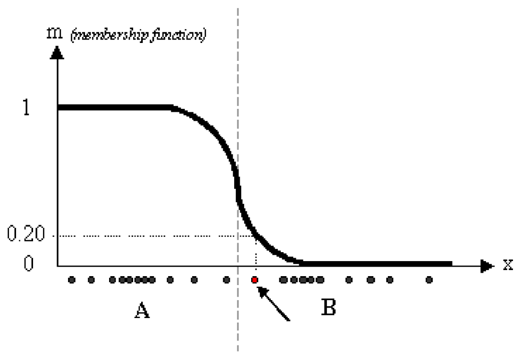
A felkészülés, a használt eszközök és az ajánlott irodalom

Mérésre felkészüléshez tanulmányozni kell a Beágyazott Intelligens Rendszerek (vimim137) idevágó előadásanyagát [1], ahonnan merítünk néhány alapvető módszert. A mért adatokban rejlő és a fuzzy tagsági függvények építéséhez szükséges adatcsoportosulásokat **fuzzy c-means** módszerrel fogjuk vizsgálni [3]. Valós adatokból fuzzy szabályokat **WM** (**Wang-Mendel**) módszer [2] egy lebutított változatával fogjuk származtatni.

Fuzzy klaszterezés

Normál klaszterezésnél a mérési pontokat szigorúan diszjunkt osztályokba soroljuk (egy adat egyetlenegy osztályhoz tartozik). Fuzzy klaszterezés (fuzzy c-means) a fuzzy tagsági függvények mintájára feltételezi, hogy alapja lehet annak, hogy egy mérési pont több osztályhoz is tartozzon, természetesen eltérő súllyal (tagsággal).

Az enyhén átlapolódó fuzzy klaszterek kialakítása egyszerű iteratív eljárás, mely végeztével előállnak az egyes klaszterekre jellemző súlyfüggvények. Egy mérési pont így több klaszterhez tartozik, ám a „fő” klasztere az, amihez tartozó súlya a legnagyobb.
Fuzzy c-means módszert Fuzzy Toolbox fcm.m függvény valósítja meg (itt a Fuzzy-meres-labor-2013.zip csomagba átmásolva). Konkrét mérési adatok mellett lehetőséget kapunk benne kísérletezni a kialakítandó klaszterek számával és az átlapolódás mértékével (az un. kitevő).



Egy fuzzy klaszter (A) súlyfüggvénye

A kialakított klaszter súlyfüggvények tulajdonképpen már legális fuzzy tagsági függvények (a [0,1] intervallumba eső értékükkel), az alakjuk azonban irreguláris és a megadásukhoz az értékeiket pontonként fel kellene sorolni. A mérésnél foglalkozunk majd azzal, hogy az ilyen irreguláris tagsági függvényeket a szokványos, könnyen paraméterezhető Gauss, háromszög, ill. trapéz alakra leegyszerűsítsük, a későbbi egyszerűbb használatuk érdekében.

Fuzzy szabályok mért adatokból – BIR labor mérési utasítása

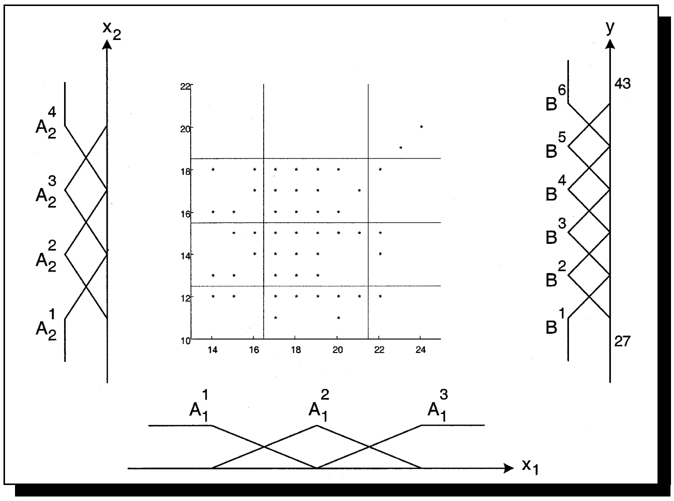
Fuzzy szabályok generálása mért adatokból

Ha a mérési adatokra jellemző fizikai univerzumokat lefedtük már a kialakított fuzzy tagsági függvényekkel, léphetünk tovább és kísérletezhetünk fuzzy szabályok kialakításával, felhasználva a mért adatokban rejlő információt. Az ötlet abból áll, hogy a kellő számú megfigyelt, manuális szabályozásra jellemző adatból kigenerált (megtanult), redundancia mentes fuzzy szabályhalmaz egy tömör, absztrakt prediktív modell, amit felhasználhatunk a megfigyelt manuális szabályozás automatizálásához. Az alkalmazott módszer az un. WM (Wang-Mendel, [2]) módszer, amiből a méréshez csak a legszükségesebb elemeket emeltünk át (leegyszerűsített klaszterezés és a konklúzió számítása).

Ha az elképzelt manuális szabályozás sémája:

IF szenzor1 ilyen, meg ilyen AND
szenzor2 ilyen, meg ilyen AND
....
szenzorN ilyen, meg ilyen
THEN a beavatkozó ilyen, meg ilyen,

akkor a bemeneti univerzumokon létesített fuzzy tagsági függvények az univerzumok Descartes szorzatán cellarendszert alakítanak ki:



A bemeneti univerzumtér cella rendszere

A cellákba eső pontok az univerzumbeli értékei alapján az oda eső jelkombinációkat jelzik. Egy-egy cella egy-egy konkrét fuzzy premisszát jelent, pl. a bal alsó cella az: ($X1 = A_1^1$ AND $X2 = A_2^1$) feltételnek felel meg. A cellák adatokkal való kitöltése így lehetőséget ad megállapítani, mely premisszát kifejezetten érdemes a végleges szabályhalmazba beépíteni. Kérdéses marad még a fuzzy szabályok következmény része. Ott is meg kell vizsgálni, hogy a kimeneti univerzumon létesített fuzzy tagsági függvényekből (jobboldal) melyek adják meg a kimeneti (beavatkozás) adatra a legnagyobb tagsági függvény értéket.

A felkészülés, az ellenőrző kérdések és a puskák

1. A mérésnél először teljesen mű adatokon begyakoroljuk a legfontosabb algoritmikus lépéseket. Ehhez sorra le kell futtatni és elemezni a mérési feladatokhoz tartozó bekeretezett kódrészleteket (megtalálható külön a Fuzzy-meres-labor-2013.zip csomagban).

2. A méréshez tartozó „puska" fájlrészletek elemzéséből látszik, hogy a fuzzy szabályok tanulásánál a konklúzió kimeneti fuzzy halmaz azonosítása igen leegyszerűsített, ami nagyobb hibák forrása is lehet (azonos premisszával, de eltérő konklúzióhoz vezető, konfliktusban lévő szabályok kezelése). Találja meg a kódban az idézett részletet és tegyen javaslatot egy olyan kódrészletre, ami pontosabban, de a mellékelt mérésadat-feldolgozáshoz illeszkedően követné a WM módszer ajánlását (konfliktuscsoporthoz tartozó szabályok átlagos kimenete).

3. Ha sikeresen megtekintjük a feldolgozás lépéseit, ideje neki rugaszkodni élet hűbb adatoknak. A tényleges mérési adatok helyett egy olyan szimulált környezetben begyűjtött adatokkal fogunk dolgozni, ahol a szoba belsőhőmérsékletének az érzete és az ablakon kívüli zord, vagy éppen kánikula idő, befolyásolják, hogy a szoba lakói hogyan tartják nyitva az ablakot. A rendszer célja kitanulni az emberek preferenciáit és az eltárolt fuzzy szabályok felhasználásával mentesíteni őket az ablak kezelésétől (feltéve, hogy a rendszer az ablak kezelését tényleg jól tanulta meg).

Fuzzy szabályok mért adatokból – BIR labor mérési utasítása

Ellenőrző kérdések:

- Mi a fuzzy c-means módszer vázlatos működése?
- Mi a kitevő és a klaszterezés kapcsolata?
- Mi egy fuzzy szabályozó rendszer felépítése?
- Hogyan néz ki egy fuzzy szabály és mi az értelmezése (kiértékelése)?
- Mi a W-M módszer lényege (képletek nyomán informálisan)?

A mérés

A mérés előkészítése

A mérés előkészítéséhez tartozik a szimulált mintapont-halmazok generálása, elképzelt bemeneti és kimeneti univerzumokon. A szimulált kísérletezéshez minden bemeneti univerzum egyforma lehet, azonban a későbbi, a tényleges mérésekre támaszkodó feldolgozásnál figyelembe kell venni az egyes univerzumoknak a szenzoroktól függő értékterjedelmét. Futtassuk le:

birlab.m kódja ld. az útmutató végén

puska0sz.m

```
% A meres elokeszitese

clear all;                           % minden valtozo torlese
close all;                           % minden abra torlese
rand('twister',sum(100*clock));      % veletlen szamgenerator veletlen
                                     % inicializalasa

% Szimulalt adatok generalasa egy db bemeneti univerzumhoz

load birlabdata;

N = 500;   % adatpontok szama
data1 = datav(1,:)';
data2 = datav(2,:)';
datay = datav(3,:)';
```

1. Feladat: Kísérletezés fuzzy klaszterezéssel

Az előkészített adatokat fuzzy módon klaszterezzük. Kísérletekben állítsunk be különböző klaszterszámokat és kitevőértékeket. Figyeljük meg, hogy mennyire sikeres a klaszterezés, ha a beállított klaszterszám megfelel, ill. nem felel meg az adatokban rejlő természetes klaszterszámnak. Figyeljük azt is, hogy a klaszter fuzzy súly alakja hogyan függ a kitevő értékétől.

puska11sz.m, **puska12sz.m**, **puska1ysz.m**

```
% 1. Feladat: Kiserletezes fuzzy klaszterezessel

% 1D (egy darab bemeneti univerzum) adatok klaszterezese,
% kulonbozo kiserletek, irodalom: Clustering - Fuzzy C-means.mht

NCl=3;                      % klaszterszam hatasa
Exponent=1.2;               % kitevo hatasa, pl. 1 - teglalap,
                            % 1.2 - kb. trapez, 1.5 - kb. Gauss
```

```
% fuzzy klaszterezes hivasa

[centers,U,obj_fcn] = fcm(data1,NCl,Exponent);
                             % centers - a klaszterek kozeppontja
                             % U a klaszterpontok sulya

figure(1), plot(data1,U,'o'); grid;
title('Fuzzy klaszterek'); xlabel('osszes adat');

% klaszter sulyok elkulonitese klaszterenkent
% egy bemeneti pont tenyleges klasztere, amely a tobbinel nagyobb
% sulyt rendel a bemeneti ponthoz

maxU = max(U);                       % adat klasztere, ra nezve max tagsagu
index1 = find(U(1,:) == maxU);       % adatok, tagsaga klaszter 1-ben max
index2 = find(U(2,:) == maxU);       % adatok, tagsaga klaszter 2-ben max
index3 = find(U(3,:) == maxU);       % adatok, tagsaga klaszter 3-ben max

figure(2), subplot(3,1,1), plot(data1(index1),U(1,index1),'og');
grid; title('Fuzzy klaszter 1, sajat adatai felett');
xlabel('klaszeter 1 adatai');
figure(2), subplot(3,1,2), plot(data1(index2),U(2,index2),'ob');
grid; title('Fuzzy klaszter 2, sajat adatai felett');
xlabel('klaszter 2 adatai');
figure(2), subplot(3,1,3), plot(data1(index3),U(3,index3),'or');
grid; title('Fuzzy klaszter 3, sajat adatai felett');
xlabel('klaszter 3 adatai');

xdata1=data1'; ydata11=U(1,:); ydata12=U(2,:); ydata13=U(3,:); % atnevezes

figure(4), subplot(3,1,1), plot(xdata1, ydata11,'og');
grid; title('Fuzzy klaszter 1 sulyfuggvenye');
xlabel('osszes adaterjedelem');
figure(4), subplot(3,1,2), plot(xdata1, ydata12,'ob');
grid; title('Fuzzy klaszter 2 sulyfuggvenye');
xlabel('osszes adaterjedelem');
figure(4), subplot(3,1,3), plot(xdata1, ydata13,'or');
grid; title('Fuzzy klaszter 2 sulyfuggvenye');
xlabel('osszes adaterjedelem');

%------------------------------------------------------
% 1. Feladat: Kiserletezes fuzzy klaszterezessel

% 1D (egy darab bemeneti univerzum) adatok klaszterezese,
% kulonbozo kiserletek, irodalom: Clustering - Fuzzy C-means.mht

NC2=3;                       % klaszterszam hatasa
Exponent=1.2;                % kitevo hatasa, pl. 1 - teglalap,
                             % 1.2 - kb. trapez, 1.5 - kb. Gauss
% fuzzy klaszterezes hivasa

[centers,U,obj_fcn] = fcm(data2,NC2,Exponent);
                             % centers - a klaszterek kozeppontja
                             % U a klaszterpontok sulya

figure(5), plot(data2,U,'o'); grid;
title('Fuzzy klaszterek'); xlabel('osszes adat');

% klaszter sulyok elkulonitese klaszterenkent
% egy bemeneti pont tenyleges klasztere, amely a tobbinel nagyobb
% sulyt rendel a bemeneti ponthoz

maxU = max(U);                       % adat klasztere, ra nezve max tagsagu
index1 = find(U(1,:) == maxU);       % adatok, tagsaga klaszter 1-ben max
```

```
index2 = find(U(2,:) == maxU);       % adatok, tagsaga klaszter 2-ben max
index3 = find(U(3,:) == maxU);       % adatok, tagsaga klaszter 3-ben max

figure(6), subplot(3,1,1), plot(data2(index1),U(1,index1),'og');
grid; title('Fuzzy klaszter 1, sajat adatai felett');
xlabel('klaszeter 1 adatai');
figure(6), subplot(3,1,2), plot(data2(index2),U(2,index2),'ob');
grid; title('Fuzzy klaszter 2, sajat adatai felett');
xlabel('klaszter 2 adatai');
figure(6), subplot(3,1,3), plot(data2(index3),U(3,index3),'or');
grid; title('Fuzzy klaszter 3, sajat adatai felett');
xlabel('klaszter 3 adatai');

xdata2=data2'; ydata21=U(1,:); ydata22=U(2,:); ydata23=U(3,:); % atnevezes

figure(7), subplot(3,1,1), plot(xdata2, ydata21,'og');
grid; title('Fuzzy klaszter 1 sulyfuggvenye');
xlabel('osszes adaterjedelem');
figure(7), subplot(3,1,2), plot(xdata2, ydata22,'ob');
grid; title('Fuzzy klaszter 2 sulyfuggvenye');
xlabel('osszes adaterjedelem');
figure(7), subplot(3,1,3), plot(xdata2, ydata23,'or');
grid; title('Fuzzy klaszter 3 sulyfuggvenye');
xlabel('osszes adaterjedelem');

%------------------------------------------------------
% 1. Feladat: Kiserletezes fuzzy klaszterezessel
% 1D (egy darab kimeneti univerzum) adatok klaszterezese,

CN=4;                % klaszterszam hatasa
ExN=1.2;             % kitevo hatasa, 1 - szogletes,
                     % 1.2 - kb. trapez, 1.5 - kb. Gauss

[centery,Uy,obj_fcn] = fcm(datay,CN,ExN);
                     % centery - a kimeneti klaszterek kozeppontja
                     % U a klaszterpontok sulya

figure(1), plot(datay,Uy,'o'); grid;
title('Kimeneti fuzzy klaszterek'); xlabel('osszes adat');
%

maxUy = max(Uy);                     % egy adathoz tartozo klaszter
                                     % a ra nezve max tagsagu
index1y = find(Uy(1,:) == maxUy);
                  % kimeneti adatok, tagsaga klaszter 1-ben max
index2y = find(Uy(2,:) == maxUy);
                  % kimeneti adatok, tagsaga klaszter 2-ben max
index3y = find(Uy(3,:) == maxUy);
                  % kimeneti adatok, tagsaga klaszter 3-ben max
index4y = find(Uy(4,:) == maxUy);
                  % kimeneti adatok, tagsaga klaszter 4-ben max

figure(9), subplot(4,1,1), plot(datay(index1y),Uy(1,index1y),'og');
grid; title('Kimeneti fuzzy klaszter 1, sajat adatai felett');
xlabel('kimeneti klaszeter 1 adatai');
figure(9), subplot(4,1,2), plot(datay(index2y),Uy(2,index2y),'ob');
grid; title('Kimeneti fuzzy klaszter 2, sajat adatai felett');
xlabel('kimeneti klaszeter 2 adatai');
figure(9), subplot(4,1,3), plot(datay(index3y),Uy(3,index3y),'ob');
grid; title('Kimeneti fuzzy klaszter 3, sajat adatai felett');
xlabel('kimeneti klaszeter 3 adatai');
figure(9), subplot(4,1,4), plot(datay(index4y),Uy(4,index4y),'ob');
grid; title('Kimeneti fuzzy klaszter 4, sajat adatai felett');
xlabel('kimeneti klaszeter 4 adatai');
```

```
xdatay=datay'; ydata1y=Uy(1,:); ydata2y=Uy(2,:);    % atnevezes
ydata3y=Uy(3,:); ydata4y=Uy(4,:);

figure(11), subplot(4,1,1), plot(xdatay, ydata1y,'o');
grid; title('Kimeneti fuzzy klaszter 1 sulyfuggvenye');
xlabel('osszes adaterjedelem');
figure(11), subplot(4,1,2), plot(xdatay, ydata2y,'o');
grid; title('Kimeneti fuzzy klaszter 2 sulyfuggvenye');
xlabel('osszes adaterjedelem');
figure(11), subplot(4,1,3), plot(xdatay, ydata3y,'o');
grid; title('Kimeneti fuzzy klaszter 3 sulyfuggvenye');
xlabel('osszes adaterjedelem');
figure(11), subplot(4,1,4), plot(xdatay, ydata4y,'o');
grid; title('Kimeneti fuzzy klaszter 4 sulyfuggvenye');
xlabel('osszes adaterjedelem');
```

2. Feladat: Fuzzy tagsági függvények kialakítása fuzzy klaszterekből

Ahogy szó volt korábban, a klaszterezésből adódó súlyok legális fuzzy tagsági függvények, csak sajnos kezelhetetlenek. Jó lenne olyan közelítésüket megadni, amik kevés paraméterrel, egyszerű függvényalakkal rendelkeznek. Ilyenek pl. a Gauss, a háromszög, ill. a trapéz tagsági függvények. A Matlab **`lsqcurvefit`** függvény (Optim Toolbox) szolgáltatásait felhasználva legkisebb négyzetes értelemben ráilleszthetjük a nyers tagsági értékekre a kívánt függvényalakot.

Kísérletezzük ki ezt mind a három javasolt függvényalakkal, a nyers adatokból közvetlenül, ill. az egyes függvénytípusokat egymásba konvertálva (mennyire releváns itt az 1. feladatban állítható kitevő?).

Megjegyzés:
Amennyire az adott gépen nincs ráinstallálva az Optim Toolbox, így a klaszterekhez tartozó, a klaszter tagsági függvényre rásimuló trapéz alakú tagsági függvények számítását más módszerrel (vesse ide be a kreativitását) oldja meg!

puska21sz.m, puska22sz.m, puska2ysz.m, puska2fajlsz.m

```
% 2. Feladat: Fuzzy tagsagi fuggvenyek kialakítasa fuzzy klaszterekbol

% a klaszter ponthalmazok kozepertek es variancia adatai
% LSE parameteres approximaciohoz
xmean11=sum(xdata1.*ydata11)/sum(ydata11);
xstd11=sum(((xdata1-xmean11).^2).*ydata11)/sum(ydata11);   % 1 klaszter
xmean12=sum(xdata1.*ydata12)/sum(ydata12);
xstd12=sum(((xdata1-xmean12).^2).*ydata12)/sum(ydata12);   % 2 klaszter
xmean13=sum(xdata1.*ydata13)/sum(ydata13);
xstd13=sum(((xdata1-xmean13).^2).*ydata13)/sum(ydata13);   % 3 klaszter

% xstd11=sqrt(xstd11); xstd12=sqrt(xstd12); xstd13=sqrt(xstd13);

% klaszter sulyfuggvenyek LSE kozelitese Gauss gorbekkel
% optimalis parameterek
xg1 = lsqcurvefit(@(xg1,xdata1) exp(-(xdata1-
xg1(1)).^2/2/xg1(2)/xg1(2)), ...
    [xmean11 xstd11], xdata1, ydata11);
% optimalis kozelito Gauss fuggveny
gfv1 = exp(-(xdata1-xg1(1)).^2/2/xg1(2)/xg1(2));

xg2 = lsqcurvefit(@(xg2,xdata1) exp(-(xdata1-
xg2(1)).^2/2/xg2(2)/xg2(2)), ...
```

```
    [xmean12 xstd12], xdata1, ydata12);
gfv2 = exp(-(xdata1-xg2(1)).^2/2/xg2(2)/xg2(2));

xg3 = lsqcurvefit(@(xg3,xdata1) exp(-(xdata1-
xg3(1)).^2/2/xg3(2)/xg3(2)), ...
    [xmean13 xstd13], xdata1, ydata13);
gfv3 = exp(-(xdata1-xg3(1)).^2/2/xg3(2)/xg3(2));

figure(5), subplot(3,1,1), plot(xdata1,ydata11,'og',xdata1,gfv1,'ro');
grid; title('1 klaszter sulyfuggvenye es optimalis Gauss kozelitese');
xlabel('osszes adaterjedelem');
figure(5), subplot(3,1,2), plot(xdata1,ydata12,'og',xdata1,gfv2,'ro');
grid; title('2 klaszter sulyfuggvenye es optimalis Gauss kozelitese');
xlabel('osszes adaterjedelem');
figure(5), subplot(3,1,3), plot(xdata1,ydata13,'og',xdata1,gfv3,'ro');
grid; title('3 klaszter sulyfuggvenye es optimalis Gauss kozelitese');
xlabel('osszes adaterjedelem');

% klaszter sulyfuggvenyek LSE kozelitese haromszog gorbekkel
% optimalis parameterek
xr1 = lsqcurvefit(@trianglefmf, [xmean11-xstd11 xmean11
xmean11+xstd11], ...
    xdata1, ydata11);
% optimalis kozelito haromszog fuggveny
tfv1 = trianglefmf(xr1,xdata1);

xr2 = lsqcurvefit(@trianglefmf, [xmean12-xstd12 xmean12
xmean12+xstd12], ...
    xdata1, ydata12);
tfv2 = trianglefmf(xr2,xdata1);

xr3 = lsqcurvefit(@trianglefmf, [xmean13-xstd13 xmean13
xmean13+xstd13], ...
    xdata1, ydata13);
tfv3 = trianglefmf(xr3,xdata1);

figure(7), subplot(3,1,1), plot(xdata1,ydata11,'og',xdata1,tfv1,'ro');
grid; title('1 klaszter sulyfuggvenye es optimalis haromszog kozelitese');
xlabel('osszes adaterjedelem');
figure(7), subplot(3,1,2), plot(xdata1,ydata12,'og',xdata1,tfv2,'ro');
grid; title('2 klaszter sulyfuggvenye es optimalis haromszog kozelitese');
xlabel('osszes adaterjedelem');
figure(7), subplot(3,1,3), plot(xdata1,ydata13,'og',xdata1,tfv3,'ro');
grid; title('3 klaszter sulyfuggvenye es optimalis haromszog kozelitese');
xlabel('osszes adaterjedelem');

% klaszter sulyfuggvenyek LSE kozelitese trapez gorbekkel
% optimalis parameterek
xtr1 = lsqcurvefit(@trapezfmf, ...
    [xmean11-2*xstd11 xmean11-xstd11 xmean11+xstd11 xmean11+2*xstd11],
xdata1, ydata11);
% optimalis kozelito trapez fuggveny
trfv1 = trapezfmf(xtr1,xdata1);

xtr2 = lsqcurvefit(@trapezfmf, ...
    [xmean12-2*xstd12 xmean12-xstd12 xmean12+xstd12 xmean12+2*xstd12],
xdata1, ydata12);
trfv2 = trapezfmf(xtr2,xdata1);

xtr3 = lsqcurvefit(@trapezfmf, ...
    [xmean13-2*xstd13 xmean13-xstd13 xmean13+xstd13 xmean13+2*xstd13],
xdata1, ydata13);
trfv3 = trapezfmf(xtr3,xdata1);
```

```
figure(8), subplot(3,1,1), plot(xdata1,ydata11,'og',xdata1,trfv1,'ro');
grid; title('1 klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');
figure(8), subplot(3,1,2), plot(xdata1,ydata12,'og',xdata1,trfv2,'ro');
grid; title('2 klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');
figure(8), subplot(3,1,3), plot(xdata1,ydata13,'og',xdata1,trfv3,'ro');
grid; title('3 klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');

% tagsagi fuggvenyek megfelelo rendezese az univerzumok menten
% bemeneti tagsagi fuggvenyek
xu1=[xtr3; xtr2; xtr1;];
xmu1=mean(xu1'); [xmu1,im]=sort(xmu1); xu1sorted=xu1(im,:);

%-----------------------------------------------------------------
% 2. Feladat: Fuzzy tagsagi fuggvenyek kialakítasa fuzzy klaszterekbol

% a klaszter ponthalmazok kozepertek es variancia adatai
% LSE parameteres approximaciohoz
xmean21=sum(xdata2.*ydata21)/sum(ydata21);
xstd21=sum(((xdata2-xmean21).^2).*ydata21)/sum(ydata21);   % 1 klaszter
xmean22=sum(xdata2.*ydata22)/sum(ydata22);
xstd22=sum(((xdata2-xmean22).^2).*ydata22)/sum(ydata22);   % 2 klaszter
xmean23=sum(xdata2.*ydata23)/sum(ydata23);
xstd23=sum(((xdata2-xmean23).^2).*ydata23)/sum(ydata23);   % 3 klaszter

% klaszter sulyfuggvenyek LSE kozelitese Gauss gorbekkel
% optimalis parameterek
xg1 = lsqcurvefit(@(xg1,xdata2) exp(-(xdata2-
xg1(1)).^2/2/xg1(2)/xg1(2)), ...
    [xmean21 xstd21], xdata2, ydata21);
% optimalis kozelito Gauss fuggveny
gfv1 = exp(-(xdata2-xg1(1)).^2/2/xg1(2)/xg1(2));

xg2 = lsqcurvefit(@(xg2,xdata2) exp(-(xdata2-
xg2(1)).^2/2/xg2(2)/xg2(2)), ...
    [xmean22 xstd22], xdata2, ydata22);
gfv2 = exp(-(xdata2-xg2(1)).^2/2/xg2(2)/xg2(2));

xg3 = lsqcurvefit(@(xg3,xdata2) exp(-(xdata2-
xg3(1)).^2/2/xg3(2)/xg3(2)), ...
    [xmean23 xstd23], xdata2, ydata23);
gfv3 = exp(-(xdata2-xg3(1)).^2/2/xg3(2)/xg3(2));

figure(5), subplot(3,1,1), plot(xdata2,ydata21,'og',xdata2,gfv1,'ro');
grid; title('1 klaszter sulyfuggvenye es optimalis Gauss kozelitese');
xlabel('osszes adaterjedelem');
figure(5), subplot(3,1,2), plot(xdata2,ydata22,'og',xdata2,gfv2,'ro');
grid; title('2 klaszter sulyfuggvenye es optimalis Gauss kozelitese');
xlabel('osszes adaterjedelem');
figure(5), subplot(3,1,3), plot(xdata2,ydata23,'og',xdata2,gfv3,'ro');
grid; title('3 klaszter sulyfuggvenye es optimalis Gauss kozelitese');
xlabel('osszes adaterjedelem');

% klaszter sulyfuggvenyek LSE kozelitese haromszog gorbekkel
% optimalis parameterek
xr1 = lsqcurvefit(@trianglefmf, [xmean21-xstd21 xmean21
xmean21+xstd21], ...
    xdata2, ydata21);
% optimalis kozelito haromszog fuggveny
tfv1 = trianglefmf(xr1,xdata2);
```

```
xr2 = lsqcurvefit(@trianglefmf, [xmean22-xstd22 xmean22
xmean22+xstd22], ...
    xdata2, ydata22);
tfv2 = trianglefmf(xr2,xdata2);

xr3 = lsqcurvefit(@trianglefmf, [xmean23-xstd23 xmean23
xmean23+xstd23], ...
    xdata2, ydata23);
tfv3 = trianglefmf(xr3,xdata2);

figure(7), subplot(3,1,1), plot(xdata2,ydata21,'og',xdata2,tfv1,'ro');
grid; title('1 klaszter sulyfuggvenye es optimalis haromszog kozelitese');
xlabel('osszes adaterjedelem');
figure(7), subplot(3,1,2), plot(xdata2,ydata22,'og',xdata2,tfv2,'ro');
grid; title('2 klaszter sulyfuggvenye es optimalis haromszog kozelitese');
xlabel('osszes adaterjedelem');
figure(7), subplot(3,1,3), plot(xdata2,ydata23,'og',xdata2,tfv3,'ro');
grid; title('3 klaszter sulyfuggvenye es optimalis haromszog kozelitese');
xlabel('osszes adaterjedelem');

% klaszter sulyfuggvenyek LSE kozelitese trapez gorbekkel
% optimalis parameterek
xtr1 = lsqcurvefit(@trapezfmf, ...
    [xmean21-2*xstd21 xmean21-xstd21 xmean21+xstd21 xmean21+2*xstd21],
xdata2, ydata21);
% optimalis kozelito trapez fuggveny
trfv1 = trapezfmf(xtr1,xdata2);

xtr2 = lsqcurvefit(@trapezfmf, ...
    [xmean22-2*xstd22 xmean22-xstd22 xmean22+xstd22 xmean22+2*xstd22],
xdata2, ydata22);
trfv2 = trapezfmf(xtr2,xdata2);

xtr3 = lsqcurvefit(@trapezfmf, ...
    [xmean23-2*xstd23 xmean23-xstd23 xmean23+xstd23 xmean23+2*xstd23],
xdata2, ydata23);
trfv3 = trapezfmf(xtr3,xdata2);

figure(8), subplot(3,1,1), plot(xdata2,ydata21,'og',xdata2,trfv1,'ro');
grid; title('1 klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');
figure(8), subplot(3,1,2), plot(xdata2,ydata22,'og',xdata2,trfv2,'ro');
grid; title('1 klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');
figure(8), subplot(3,1,3), plot(xdata2,ydata23,'og',xdata2,trfv3,'ro');
grid; title('1 klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');

% tagsagi fuggvenyek megfelelo rendezese az univerzumok menten
% bemeneti tagsagi fuggvenyek
xu2=[xtr3; xtr2; xtr1;];
xmu2=mean(xu2'); [xmu2,im]=sort(xmu2); xu2sorted=xu2(im,:);

%-------------------------------------------------------------------
% 2. Feladat: Fuzzy tagsagi fuggvenyek kialakítasa fuzzy klaszterekbol

% a kimeneti klaszter ponthalmazok kozepertek es variancia
% adatai LSE parameteres approximaciohoz
xmean1y=sum(xdatay.*ydata1y)/sum(ydata1y);
xstd1y=sum(((xdatay-xmean1y).^2).*ydata1y)/sum(ydata1y);
xstd1y=sqrt(xstd1y);
xmean2y=sum(xdatay.*ydata2y)/sum(ydata2y);
xstd2y=sum(((xdatay-xmean2y).^2).*ydata2y)/sum(ydata2y);
xstd2y=sqrt(xstd2y);
```

```
xmean3y=sum(xdatay.*ydata3y)/sum(ydata3y);
xstd3y=sum(((xdatay-xmean3y).^2).*ydata3y)/sum(ydata3y);
xstd3y=sqrt(xstd3y);
xmean4y=sum(xdatay.*ydata4y)/sum(ydata4y);
xstd4y=sum(((xdatay-xmean4y).^2).*ydata4y)/sum(ydata4y);
xstd4y=sqrt(xstd4y);

% klaszter sulyfuggvenyek LSE kozelitese Gauss gorbekkel
% optimalis parameterek
xg1y = lsqcurvefit(@(xg1y,xdatay) exp(-(xdatay-
xg1y(1)).^2/2/xg1y(2)/xg1y(2)), ...
    [xmean1y xstd1y], xdatay, ydata1y);
% optimalis kozelito Gauss fuggveny
gfv1y = exp(-(xdatay-xg1y(1)).^2/2/xg1y(2)/xg1y(2));

xg2y = lsqcurvefit(@(xg2y,xdatay) exp(-(xdatay-
xg2y(1)).^2/2/xg2y(2)/xg2y(2)), ...
    [xmean2y xstd2y], xdatay, ydata2y);
gfv2y = exp(-(xdatay-xg2y(1)).^2/2/xg2y(2)/xg2y(2));

xg3y = lsqcurvefit(@(xg3y,xdatay) exp(-(xdatay-
xg3y(1)).^2/2/xg3y(2)/xg3y(2)), ...
    [xmean3y xstd3y], xdatay, ydata3y);
gfv3y = exp(-(xdatay-xg3y(1)).^2/2/xg3y(2)/xg3y(2));

xg4y = lsqcurvefit(@(xg4y,xdatay) exp(-(xdatay-
xg4y(1)).^2/2/xg4y(2)/xg4y(2)), ...
    [xmean4y xstd4y], xdatay, ydata4y);
gfv4y = exp(-(xdatay-xg4y(1)).^2/2/xg4y(2)/xg4y(2));

figure(12), subplot(4,1,1), plot(xdatay,ydata1y,'og',xdatay,gfv1y,'ro');
grid; title('1 kimeneti klaszter sulyfuggvenye es optimalis Gauss
kozelitese');
xlabel('osszes adaterjedelem');
figure(12), subplot(4,1,2), plot(xdatay,ydata2y,'og',xdatay,gfv2y,'ro');
grid; title('2 kimeneti klaszter sulyfuggvenye es optimalis Gauss
kozelitese');
xlabel('osszes adaterjedelem');
figure(12), subplot(4,1,3), plot(xdatay,ydata3y,'og',xdatay,gfv3y,'ro');
grid; title('3 kimeneti klaszter sulyfuggvenye es optimalis Gauss
kozelitese');
xlabel('osszes adaterjedelem');
figure(12), subplot(4,1,4), plot(xdatay,ydata4y,'og',xdatay,gfv4y,'ro');
grid; title('4 kimeneti klaszter sulyfuggvenye es optimalis Gauss
kozelitese');
xlabel('osszes adaterjedelem');

% klaszter sulyfuggvenyek LSE kozelitese haromszog gorbekkel
% optimalis parameterek
xr1y = lsqcurvefit(@trianglefmf, ...
    [xmean1y-xstd1y xmean1y xmean1y+xstd1y], xdatay, ydata1y);
% optimalis kozelito haromszog fuggveny
tfv1y = trianglefmf(xr1y,xdatay);

xr2y = lsqcurvefit(@trianglefmf, ...
    [xmean2y-xstd2y xmean2y xmean2y+xstd2y], xdatay, ydata2y);
tfv2y = trianglefmf(xr2y,xdatay);

xr3y = lsqcurvefit(@trianglefmf, ...
    [xmean3y-xstd3y xmean3y xmean3y+xstd3y], xdatay, ydata3y);
tfv3y = trianglefmf(xr3y,xdatay);

xr4y = lsqcurvefit(@trianglefmf, ...
    [xmean4y-xstd4y xmean4y xmean4y+xstd4y], xdatay, ydata4y);
```

```
tfv4y = trianglefmf(xr4y,xdatay);

figure(14), subplot(4,1,1), plot(xdatay,ydata1y,'og',xdatay,tfv1y,'ro');
grid;
title('1 kimeneti klaszter sulyfuggvenye es optimalis haromszog
kozelitese');
xlabel('osszes adaterjedelem');
figure(14), subplot(4,1,2), plot(xdatay,ydata2y,'og',xdatay,tfv2y,'ro');
grid;
title('2 kimeneti klaszter sulyfuggvenye es optimalis haromszog
kozelitese');
xlabel('osszes adaterjedelem');
figure(14), subplot(4,1,3), plot(xdatay,ydata3y,'og',xdatay,tfv3y,'ro');
grid;
title('3 kimeneti klaszter sulyfuggvenye es optimalis haromszog
kozelitese');
xlabel('osszes adaterjedelem');
figure(14), subplot(4,1,4), plot(xdatay,ydata4y,'og',xdatay,tfv4y,'ro');
grid;
title('4 kimeneti klaszter sulyfuggvenye es optimalis haromszog
kozelitese');
xlabel('osszes adaterjedelem');

% klaszter sulyfuggvenyek LSE kozelitese trapez gorbekkel
% optimalis parameterek
xtr1y = lsqcurvefit(@trapezfmf, ...
    [xmean1y-xstd1y xmean1y-xstd1y/2 xmean1y+xstd1y/2 xmean1y+xstd1y],
xdatay, ydata1y);
% optimalis kozelito trapez fuggveny
trfv1y = trapezfmf(xtr1y,xdatay);

xtr2y = lsqcurvefit(@trapezfmf, ...
    [xmean2y-xstd2y xmean2y-xstd2y/2 xmean2y+xstd2y/2 xmean2y+xstd2y],
xdatay, ydata2y);
trfv2y = trapezfmf(xtr2y,xdatay);

xtr3y = lsqcurvefit(@trapezfmf, ...
    [xmean3y-xstd3y xmean3y-xstd3y/2 xmean3y+xstd3y/2 xmean3y+xstd3y],
xdatay, ydata3y);
trfv3y = trapezfmf(xtr3y,xdatay);

xtr4y = lsqcurvefit(@trapezfmf, ...
    [xmean4y-xstd4y xmean4y-xstd4y/2 xmean4y+xstd4y/2 xmean4y+xstd4y],
xdatay, ydata4y);
trfv4y = trapezfmf(xtr4y,xdatay);

figure(15), subplot(4,1,1), plot(xdatay,ydata1y,'og',xdatay,trfv1y,'ro');
grid;
title('1 kimeneti klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');
figure(15), subplot(4,1,2), plot(xdatay,ydata2y,'og',xdatay,trfv2y,'ro');
grid;
title('2 kimeneti klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');
figure(15), subplot(4,1,3), plot(xdatay,ydata3y,'og',xdatay,trfv3y,'ro');
grid;
title('3 kimeneti klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');
figure(15), subplot(4,1,4), plot(xdatay,ydata4y,'og',xdatay,trfv4y,'ro');
grid;
title('4 kimeneti klaszter sulyfuggvenye es optimalis trapez kozelitese');
xlabel('osszes adaterjedelem');
```

```
% kimeneti tagsagi fuggvenyek

xu1y=[xtr4y; xtr3y; xtr2y; xtr1y;];
xmu1y=mean(xu1y'); [xmu1y,imy]=sort(xmu1y); xu1ysorted=xu1y(imy,:);

%-----------------------------------------------------------------
% 2. Feladat: Fuzzy tagsagi fuggvenyek kialakítasa fuzzy klaszterekbol

xsorted=[xu1sorted; xu2sorted; xu1ysorted];

% tagsagi fuggveny leiro fajl elokeszitese - fuggveny parameterek
save FuzSetData.txt xsorted -ascii;

% tagsagi fuggveny leiro fajl elokeszitese - fuggveny dbszam
% univerzumonkent
fnCounts=[3 3 4];
save NumFuzSet.txt fnCounts -ascii;
```

A kialakított fuzzy tagsági függvényeknél ügyelni kell arra, hogy a klaszterezésnél keletkezett klaszterek sorszáma nem szükségképpen felel meg a természetes fizikai nagyság szerinti rendezésüknek. A tagsági függvény leírófájl helyes legenerálásához az egyes függvényeket az univerzumbeli pozíciójuknak megfelelően „sorba kell rakni". Trapéz tagsági függvények esetén a leíró formátum:

```
a11 b11 c11 d11      % az 1. bemeneti univerzum 1. tagsági függvény jellemző pontjai
a12 b12 c12 d12      % az 1. bemeneti univerzum 2. tagsági függvény jellemző pontjai
... ... ... ... ... ... ...
a1N b1N c1N d1N      % az 1. bemeneti univerzum N. tagsági függvény jellemző pontjai


a21 b21 c21 d21      % a 2. bemeneti univerzum 1. tagsági függvény jellemző pontjai
a22 b22 c22 d22      % a 2. bemeneti univerzum 2. tagsági függvény jellemző pontjai
... ... ... ... ... ... ...
a2N b2N c2N d2N      % a 2. bemeneti univerzum N. tagsági függvény jellemző pontjai
... ... ... ... ... ... ...

aK1 bK1 cK1 dK1      % a k. bemeneti univerzum 1. tagsági függvény jellemző pontjai
aK2 bK2 cK2 dK2      % a k. bemeneti univerzum 2. tagsági függvény jellemző pontjai
... ... ... ... ... ... ...
aKN bKN cKN dKN      % a k. bemeneti univerzum N. tagsági függvény jellemző pontjai

a11 b11 c11 d11      % a kimeneti univerzum 1. tagsági függvény jellemző pontjai
a12 b12 c12 d12      % a kimeneti univerzum 2. tagsági függvény jellemző pontjai
... ... ... ... ... ... ...
a1M b1M c1M d1M      % a kimeneti univerzum M. tagsági függvény jellemző pontjai
```

3. Feladat: Fuzzy szabályok tanulása mért adatokból WM módszerrel

Hasonlítsa össze a WM-módszer elvi leírását [2] a lenti minimális implementálással. A mintának megfelelően készítse el a „mért" jelszekvenciákat és állítsa elő az azokat leíró fuzzy szabályhalmazt.

puska3.m

```
% 3. Feladat: Fuzzy szabalyok tanulasa mert adatokbol WM modszerrel
(szimulalt adatok)

% fuzzy szabalyok tanulasa mert adatok alapjan, irodalom:
% WM Method Completed A Flexible Fuzzy System Approach to Data Mining.pdf

load birlabdata;
N = 500;   % adatpontok szama
%data1 = datav(1,:)';
%data2 = datav(2,:)';
%datay = datav(3,:)';

save DataTrain.txt datav -ascii;

% a szukseges adatok es leirok beolvasasa
load DataTrain.txt;           % bemeneti adatok: x1(t), x2(t), x3(t), y(t)
load NumFuzSet.txt;           % fuzzy halmazok szama univerzumonkent
load FuzSetData.txt;          % fuzzy halmazok parameterei univerzumokhoz

extra=DataTrain';             % bemeneti adatok: x1(t), x2(t), x3(t), y(t)
fnCounts=NumFuzSet;           % fuzzy halmazok szama univerzumonkent
fmfsets=FuzSetData;           % fuzzy halmazok parameterei univerzumokhoz
[NSamples,m]=size(extra);
NInput=m-1;

% Tagsagi fuggvenyek a bemeneti univerzumok Descartes szorzata felett
% cellakat definialnak (jelen esetben 9 cella van)
% Minden adatszelet [x1(t) x2(t)] valamelyik bemeneti cellahoz,
% tartozik azaz a cellat definialo tagsagi fuggvenyek az egyes x(t)
% ertekekre legnagyobb ertekeket adnak
actGradesIn=zeros(NSamples,NInput); % egy szelethez tartozo bemeneti
                                    % tagsagi fuggveny ertekek: a1, a2
actGradesOut=zeros(NSamples,1);  % egy szelethez tartozo kimeneti tagsagi
                                 % fuggveny ertek: b
actFnIn=zeros(NSamples,NInput);  % egy szelethez tartozo bemeneti tagsagi
                                 % fuggveny indexek: tf1, tf2
actFnOut=zeros(NSamples,1);      % egy szelethez tartozo kimeneti tagsagi
                                 % fuggveny index: tfy
actRulesTmp=zeros(NSamples,NInput+3);   % ideiglenes szabalyrendszerleiro
                                        % [tf1 tf2 a tfy b]

% ideiglenes szabalyrendszer szamitasa: egy szabaly bemenetei
% idoszeletenkent
for k=1:NSamples                        % idoszeletek
  for i=1:NInput                        % bemenetek
    NFns=fnCounts(i); gradeV=zeros(1,NFns);
    for nthFn=1:NFns,         % adott univerzum tagsagi fuggvenyei szerint
      grade=mebextfms(NFns,i,nthFn,extra(k,i),fmfsets); % tags. fv erteke
      gradeV(nthFn)=grade;            % eltarolva
    end;
    [gradeMax,indexMax]=max(gradeV);    % melyiknel erjuk el a maximumot?
    actGradesIn(k,i)=gradeMax; actFnIn(k,i)=indexMax; % taroljuk el
  end;
                                        % es most u.a. a kimeneti adatra
  NFnsy=fnCounts(m); NFns=fnCounts(m-1); gradeV=zeros(1,NFns);
  for nthFn=1:NFnsy,            % adott univerzum tags. fuggvenyei szerint
      grade=mebextfms(NFns,m,nthFn,extra(k,m),fmfsets); % tags. fv erteke
      gradeV(nthFn)=grade;              % eltarolva
  end;
  [gradeMax,indexMax]=max(gradeV);      % melyiknel erjuk el a maximumot?
```

```
                                         % ideiglenes leirok elokeszitese
  actGradesOut(k)=gradeMax; actFnOut(k)=indexMax;
  actRulesTmp(k,:)=[actFnIn(k,:), max(actGradesIn(k,:)), actFnOut(k),
actGradesOut(k)];
end;

actRules=[];
     % a vegleges szabalyleiro: [X1 X2 Y] (X1 AND X2 THEN Y szabalyhoz)
     % redundanciamentes modon (minden szabaly csak egyszer)

for i1=1:fnCounts(1), i1,
    for i2=1:fnCounts(2), i2,    % ciklusok bemeneti univerzumonkent
                                 % az ott definialt tagsagi fuggvenyek
                                 % szerint
            Nifindex=zeros(1,NSamples); % hol fordul elo actulResTmp-ben
                                 % az adott i1, i2 bemeneti kombinacio
            for k=1:NSamples,
              cc=(actRulesTmp(k,[1:2])==[i1 i2]); if cc(1)&cc(2),
Nifindex(k)=1; end;
            end;
            Ni=find(Nifindex==1);
            if isempty(Ni),
            else              % a szabalyleirobol csak az adott bemeneti
                              %kombinaciohoz tartozo szabalyokat
                              % valasztjuk ki
                actRulesTmp2=actRulesTmp(Ni,:); NLi=length(Ni);

                Nifindex1=[];
                for k=1:NLi, % ezekre a szabalyokra megnezzuk, hogy
                             % melyiknek eredmenye
                             % az 1. kimeneti tagsagi fuggveny
                    cc=(actRulesTmp2(k,4)==1);
                    if cc, Nifindex1=[Nifindex1 k]; end; % gyujtsuk ki
                end;
                actRulesTmpY1=actRulesTmp2(Nifindex1,:); % taroljuk el
                NLiY1=length(Nifindex1);    % hany ilyen szabaly van?
                MaxGrRY1=max(actRulesTmpY1(:,3)); % mi a max. bemeneti
                                                  % cella sulya
                MaxGrOutY1=max(actRulesTmpY1(:,5)); % mi a max. kimeneti
                                                    % tf ertek?

                Nifindex2=[];
                for k=1:NLi, cc=(actRulesTmp2(k,4)==2);
                       if cc, Nifindex2=[Nifindex2 k]; end;  % gyujtsuk ki
                end;
                actRulesTmpY2=actRulesTmp2(Nifindex2,:);
                NLiY2=length(Nifindex2);
                MaxGrRY2=max(actRulesTmpY2(:,3));
                MaxGrOutY2=max(actRulesTmpY2(:,5));

                Nifindex3=[];
                for k=1:NLi, cc=(actRulesTmp2(k,4)==3);
                     if cc, Nifindex3=[Nifindex3 k]; end;  % gyujtsuk ki
                end;
                actRulesTmpY3=actRulesTmp2(Nifindex3,:);
                NLiY3=length(Nifindex3);
                MaxGrRY3=max(actRulesTmpY3(:,3));
                MaxGrOutY3=max(actRulesTmpY3(:,5));

                Nifindex4=[];
                for k=1:NLi, cc=(actRulesTmp2(k,4)==4);
                     if cc, Nifindex4=[Nifindex4 k]; end;  % gyujtsuk ki
                end;
                actRulesTmpY4=actRulesTmp2(Nifindex4,:);
```

```
                NLiY4=length(Nifindex4);
                MaxGrRY4=max(actRulesTmpY4(:,3));
                MaxGrOutY4=max(actRulesTmpY4(:,5));

                % most dontsuk el, hogy a szabalyt 1., vagy ... 4.
                % kimeneti tagsagi fuggvennyel zarjuk
                [Ymax, Ypos]=max([MaxGrRY1*NLiY1, MaxGrRY2*NLiY2,
MaxGrRY3*NLiY3, MaxGrRY4*NLiY4]);
                Yi=Ypos;
                actRules=[actRules; [i1 i2 Yi]]; % a vegleges megtanult
                                                 % szabalyleiro
            end; % k
    end; % i2
end; % i1

save actRules.txt actRules -ascii;  % a vegleges megtanult szabalyleiro
```

4. Feladat: A megtanult szabályok verifikálása

Tesztelje meg a 3. feladatban kialakított szabályokat. E célból készítsen egy alkalmas tesztjelhalmazt (birlab.m), amit a kiszámított szabályokkal működő fuzzy következtető rendszerrel fog feldolgozni. A mintának megfelelően készítse el a „mért” jelszekvenciákat és jósolja meg fuzzy rendszerrel a kimenet értékeit. Értékelje a látottakat.
Alakítsa át a kódot, hogy a Gauss és a háromszög függvényekkel is lehessen dolgozni.

puska4sz.m

```
% 4. Feladat: A megtanult szabalyok verifikalasa (szimulalt adatok)

% a tanult fuzzy szabalyok verifikalasa kulon teszt adatok alapjan

load birlabtest;
N = 500;   % adatpontok szama
save DataTest.txt datav -ascii;

load DataTest.txt;         % tesztadatok a megszokott formatumban
load actRules.txt;         % legeneralt fuzzy szabalyok
load NumFuzSet.txt;        % fuzzy halmazok szama univerzumonkent
load FuzSetData.txt;       % fuzzy halmazok tagsagi fuggvenyei

%----------------------------------------
exapp=DataTest';
[NSamples,m]=size(exapp);
NInput=m-1;
actGrades=zeros(NSamples,4);    % az elsutott fuzzy szabalyok sulya
fmfsets=FuzSetData;
NR=length(actRules);
yp=zeros(NSamples,1);           % josolt kimeneti ertekek

yt=exapp(:,m);                  % teszt kimeneti ertekek

Ff1=fmfsets(NInput*NFns+1,:); Ff2=fmfsets(NInput*NFns+2,:);
Ff3=fmfsets(NInput*NFns+3,:); Ff4=fmfsets(NInput*NFns+4,:);
                                % a 4 kimeneti tagsagi fuggveny leiro
y=[min(exapp(:,3)):.1:max(exapp(:,3))]; % a kimeneti adatok terjedelme
mf1=trapmf(y,Ff1); mf2=trapmf(y,Ff2);   % a kimeneteti trapez
mf3=trapmf(y,Ff3); mf4=trapmf(y,Ff4);   % tagsagi fuggvenyek generalasa

for k=1:NSamples            % idoszeletenkent
```

```
  for ir=1:NR, FSi=actRules(ir,:);
      gradeV=zeros(1,NInput);  % szabalyonkent
      % a fuzzy rendszer mindig minden szabalyt ertekel ki lepesenkent
   for i=1:NInput,
    x=exapp(k,i);           % egy bemenethez tartozo adat
    NFns=fnCounts(i); grade=mebextfms(NFns,i,FSi(i),x,fmfsets);
                             % es fuzzy sulya
    gradeV(i)=grade;
   end;
   gradeMin=min(gradeV);    % fuzzy szab. premissza sulya (minmax kovetk.)
   if actGrades(k,FSi(3))<gradeMin,
       actGrades(k,FSi(3))=gradeMin;
                             % az elsutheto szab-k kozul a max. sulyu
   end;
  end;
  if actGrades(k,1)+actGrades(k,2)+actGrades(k,3)+actGrades(k,4)>0,
                        % ha van nem 0 szintu kimenetei tagsagi fuggveny
      yp(k)=defuzz(y,actGrades(k,1)*mf1+actGrades(k,2)*mf2+actGrades(k,3)*
mf3+actGrades(k,4)*mf4,'centroid');
                        % akkor annak defuzzifikalasaval a josolt ertek
  else yp(k)=0;         % alapeseti ertek hiba (szab. nelkuli cella) esete
      % igazi megoldas - a szabalyrendszzer kiterjesztese minden cellara,
      % arra az esetre, ha a tanulasnal hasznalt adatok nem minden cellaba
      % estek bele, igy nem minden cellahoz rendelten sikerult felepiteni
      % a benne ervenyes fuzzy szabalyt
  end;
end; %k

err = yt-yp; % a josolt ertek hibaja
elsum = sum(abs(err));
elav = elsum/NSamples; % atlagos abszolut hiba
errel = 100*abs(err./yp);

errave = abs(err);
for i=5:N-5, errave(i)=mean(abs(err(i-4:i+4))); end;

figure(20), subplot(311),
plot([1:NSamples],yt,'b*',[1:NSamples],yt,'b',[1:NSamples],yp,'ro',
[1:NSamples],yp,'r');
grid; title('* igazi, o model altal josolt');
subplot(312), plot([1:NSamples],abs(err),'b',[1:NSamples],errave,'r');
grid; title('hiba');
subplot(313), plot(errel); grid; title('relativ hiba');
axis([1 NSamples 0 200]);
```

5. Feladat: További kísérletek tudásbázissal

A szimulált mérési adatokat generáló **birlab.m** fájlban kísérletezzen a fuzzy halmazok alatti univerzumokban másképpen szétszórni a kérési értékeket (pl. a generáló eloszlások középpontjainak és szórásainak módosításával). A teljes procedura lefuttatásához használhatja alkalmasan módosított **teljes.m** programot.

```
birlab;      % tanulo es teszt adatbazis szamitasa
              % kesobb: birlabplus

             % datav(i,:) = [belso t, kulso t, ablaknyitas]
             % belso homerseklet =   alacsony - 19 Cfok korul
             %                       kozepes - 21 Cfok korul
```

```
            %                          magas - 24 Cfok fele
            % kulso homerseklet =      alacsony - 5 Cfok felett
            %                          kozepes - 20 Cfok korul
            %                          magas - 35 Cfok fele
            % ablaknyitas =            resnyire - majdnem 0%
            %                          kicsit - 30% korul
            %                          mertekkel - 50% korul
            %                          kitarva - majdnem 100%

puska0sz; 'puska0sz', pause;      % adat elokeszites
puska11sz; 'puska11sz', pause;    % 1. bemenet (belso t) adatainak klaszterezese
puska12sz; 'puska12sz', pause;    % 2. bemenet (kulso t) adatainak klaszterezese
puska1ysz; 'puska1ysz', pause;    % kimenet (ablaknyitas) adatainak klaszterezese
puska21sz; 'puska21sz', pause;    % tagsagi fv illesztese 1. bemeneti klaszterre
puska22sz; 'puska22sz', pause;    % tagsagi fv illesztese 2. bemeneti klaszterre
puska2ysz; 'puska2ysz', pause;    % tagsagi fv illesztese kimeneti klaszterre
puska2fajlsz; 'puska2fajlsz', pause;    % tagsagi fv leirok elkeszitese
puska3sz; 'puska3sz', pause;      % fuzzy szabalyok tanulasa tanulo adatokbol
puska4sz; 'puska4sz', pause;      % megtanult fuzzy szabalyok alkalmazasa,
                                  % osszevetese a teszt adatokkal

% es meg egy ellenorzes: actRules.txt es a birlab.m -beli TB osszehasonlitasa!
```

Kísérletezzen tovább azzal, hogy a rendszer betanítása után változik az ambiens tér ember általi kezelése (ennek egy modellje a **birlabplus.m** fájlban található). Futtassa le a korábban megtanított fuzzy szabályozó rendszer erre az esetre és értékelje a tapasztalatait.
Igyekezzen önálló ötletekkel gazdagítani a kísérletezést.

Ajánlott irodalom:

[1] Beágyazott intelligens rendszerek: előadásjegyzet: „kontroll-tanulas” előadások
[2] The WM Method Completed A Flexible Fuzzy System Approach to Data Mining.pdf (a fuzzy-meres-fajlok könyvtárban)
[3] Clustering - Fuzzy C-means,
http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/cmeans.html

Függvények

```
function F=trapezfmf(x,xdata)
N=length(xdata); F=zeros(1,N);
for i=1:N, if xdata(i)<x(1), F(i)=0;
           elseif xdata(i)<x(2), F(i)=(xdata(i)-x(1))/(x(2)-x(1));
           elseif xdata(i)<x(3), F(i)=1;
           elseif xdata(i)<x(4), F(i)=(x(4)-xdata(i))/(x(4)-x(3));
           else F(i)=0; end;
end;
```

```
function F=trianglefmf(x,xdata)
N=length(xdata); F=zeros(1,N);
for i=1:N, if xdata(i)<x(1), F(i)=0;
           elseif xdata(i)<x(2), F(i)=(xdata(i)-x(1))/(x(2)-x(1));
           elseif xdata(i)<x(3), F(i)=(x(3)-xdata(i))/(x(3)-x(2));
           else F(i)=0; end;
end;
```

```
function y=rndgen(ss,m,s)
y1 = m+s*randn;
if (ss=='L')&(y1<m), y = 2*m-y1;
elseif (ss=='R')&(y1>m), y = 2*m-y1;
else y = y1; end;
```

```
%if y <0, ss, disp([y1, m, s, y]); end;
```

```
function [U_new, center, obj_fcn] = stepfcm(data, U, cluster_n, expo)
%STEPFCM One step in fuzzy c-mean clustering.
%   [U_NEW, CENTER, ERR] = STEPFCM(DATA, U, CLUSTER_N, EXPO)
%   performs one iteration of fuzzy c-mean clustering, where
%
%   DATA: matrix of data to be clustered. (Each row is a data point.)
%   U: partition matrix. (U(i,j) is the MF value of data j in cluster j.)
%   CLUSTER_N: number of clusters.
%   EXPO: exponent (> 1) for the partition matrix.
%   U_NEW: new partition matrix.
%   CENTER: center of clusters. (Each row is a center.)
%   ERR: objective function for partition U.
%
%   Note that the situation of "singularity" (one of the data points is
%   exactly the same as one of the cluster centers) is not checked.
%   However, it hardly occurs in practice.
%
%       See also DISTFCM, INITFCM, IRISFCM, FCMDEMO, FCM.

%   Roger Jang, 11-22-94.
%   Copyright 1994-2002 The MathWorks, Inc.
%   $Revision: 1.13 $  $Date: 2002/04/14 22:21:02 $

mf = U.^expo;       % MF matrix after exponential modification
center = mf*data./((ones(size(data, 2), 1)*sum(mf'))'); % new center
dist = distfcm(center, data);       % fill the distance matrix
obj_fcn = sum(sum((dist.^2).*mf));  % objective function
tmp = dist.^(-2/(expo-1));      % calculate new U, suppose expo != 1
U_new = tmp./(ones(cluster_n, 1)*sum(tmp));
```

```
function y = mebextfms(N,nofinput,ithfs,xdatapoint,fmfsets)
%
% nofinput       - bemeneti univerzum indexe, ill. a kimeneti univerzum indexe
%                   (bemenetek szama + 1)
% N             - fuzzy halmazok szama egy univerzum felett
% ithfs          - fuzzy halmaz indexe (meresnel 1..3 bemenet, 1..4 kimenet)
% xdatapoint     - mert adat
% fmfsets        - fuzzy halmaz leiro, ld. kesobb
%
%  a11, b11, c11, d11
%  a12, b12, c12, d12
%    ...
%  a1N, b1N, c1N, d1N    - N db trapez tagsagi fuggveny az 1. bemenethez
%  a21, b21, c21, d21
%  a22, b22, c22, d22
%    ...
%  a2N, b2N, c2N, d2N    - N db trapez tagsagi fuggveny a 2. bemenethez
%    stb. stb.
%  a11, b11, c11, d11
%  a12, b12, c12, d12
%    ...
%  a1N1, b1N1, c1N1, d1N1 - N1 db trapez tagsagi fuggveny a kimenethez
%                        (meresnel tipikusan N=3, N1=2 lesz)

aindex=(nofinput-1)*N+ithfs;
a=fmfsets(aindex,1);
b=fmfsets(aindex,2);
c=fmfsets(aindex,3);
d=fmfsets(aindex,4);

if xdatapoint<a, y=0;
```

```
elseif xdatapoint<b, y=(xdatapoint-a)/(b-a);
elseif xdatapoint<c, y=1;
elseif xdatapoint<d, y=(d-xdatapoint)/(d-c);
else y=0; end;
```

```
function U = initfcm(cluster_n, data_n)
%INITFCM Generate initial fuzzy partition matrix for fuzzy c-means clustering.
%   U = INITFCM(CLUSTER_N, DATA_N) randomly generates a fuzzy partition
%   matrix U that is CLUSTER_N by DATA_N, where CLUSTER_N is number of
%   clusters and DATA_N is number of data points. The summation of each
%   column of the generated U is equal to unity, as required by fuzzy
%   c-means clustering.
%
%       See also DISTFCM, FCMDEMO, IRISFCM, STEPFCM, FCM.

%   Roger Jang, 12-1-94.
%   Copyright 1994-2002 The MathWorks, Inc.
%   $Revision: 1.11 $  $Date: 2002/04/14 22:21:58 $

U = rand(cluster_n, data_n);
col_sum = sum(U);
U = U./col_sum(ones(cluster_n, 1), :);
```

```
function [center, U, obj_fcn] = fcm(data, cluster_n, options)
%FCM Data set clustering using fuzzy c-means clustering.
%
%   [CENTER, U, OBJ_FCN] = FCM(DATA, N_CLUSTER) finds N_CLUSTER number of
%   clusters in the data set DATA. DATA is size M-by-N, where M is the number of
%   data points and N is the number of coordinates for each data point. The
%   coordinates for each cluster center are returned in the rows of the matrix
%   CENTER. The membership function matrix U contains the grade of membership of
%   each DATA point in each cluster. The values 0 and 1 indicate no membership
%   and full membership respectively. Grades between 0 and 1 indicate that the
%   data point has partial membership in a cluster. At each iteration, an
%   objective function is minimized to find the best location for the clusters
%   and its values are returned in OBJ_FCN.
%
%   [CENTER, ...] = FCM(DATA,N_CLUSTER,OPTIONS) specifies a vector of options
%   for the clustering process:
%       OPTIONS(1): exponent for the matrix U             (default: 2.0)
%       OPTIONS(2): maximum number of iterations          (default: 100)
%       OPTIONS(3): minimum amount of improvement         (default: 1e-5)
%       OPTIONS(4): info display during iteration         (default: 1)
%   The clustering process stops when the maximum number of iterations
%   is reached, or when the objective function improvement between two
%   consecutive iterations is less than the minimum amount of improvement
%   specified. Use NaN to select the default value.
%
%   Example
%       data = rand(100,2);
%       [center,U,obj_fcn] = fcm(data,2);
%       plot(data(:,1), data(:,2),'o');
%       hold on;
%       maxU = max(U);
%       % Find the data points with highest grade of membership in cluster 1
%       index1 = find(U(1,:) == maxU);
%       % Find the data points with highest grade of membership in cluster 2
%       index2 = find(U(2,:) == maxU);
%       line(data(index1,1),data(index1,2),'marker','*','color','g');
%       line(data(index2,1),data(index2,2),'marker','*','color','r');
%       % Plot the cluster centers
%       plot([center([1 2],1)],[center([1 2],2)],'*','color','k')
%       hold off;
```

```
%
%   See also FCMDEMO, INITFCM, IRISFCM, DISTFCM, STEPFCM.

%   Roger Jang, 12-13-94, N. Hickey 04-16-01
%   Copyright 1994-2002 The MathWorks, Inc.
%   $Revision: 1.13 $  $Date: 2002/04/14 22:20:38 $

if nargin ~= 2 & nargin ~= 3,
     error('Too many or too few input arguments!');
end

data_n = size(data, 1);
in_n = size(data, 2);

% Change the following to set default options
default_options = [2;   % exponent for the partition matrix U
           100;  % max. number of iteration
           1e-5; % min. amount of improvement
           1];   % info display during iteration

if nargin == 2,
     options = default_options;
else
     % If "options" is not fully specified, pad it with default values.
     if length(options) < 4,
           tmp = default_options;
           tmp(1:length(options)) = options;
           options = tmp;
     end
     % If some entries of "options" are nan's, replace them with defaults.
     nan_index = find(isnan(options)==1);
     options(nan_index) = default_options(nan_index);
     if options(1) <= 1,
           error('The exponent should be greater than 1!');
     end
end

expo = options(1);           % Exponent for U
max_iter = options(2);       % Max. iteration
min_impro = options(3);      % Min. improvement
display = options(4);        % Display info or not

obj_fcn = zeros(max_iter, 1);% Array for objective function

U = initfcm(cluster_n, data_n);                % Initial fuzzy partition
% Main loop
for i = 1:max_iter,
     [U, center, obj_fcn(i)] = stepfcm(data, U, cluster_n, expo);
     if display,
           fprintf('Iteration count = %d, obj. fcn = %f\n', i, obj_fcn(i));
     end
     % check termination condition
     if i > 1,
           if abs(obj_fcn(i) - obj_fcn(i-1)) < min_impro, break; end,
     end
end

iter_n = i;% Actual number of iterations
obj_fcn(iter_n+1:max_iter) = [];
```

birlab.m

```
clear all; close all; rand('seed',0);

tbmean = [18, 21, 24]; tbdev = [1.5, 1.5, 1.5];
tkmean = [5, 20, 35]; tkdev = [6, 6, 6];
amean = [0, 30, 50, 100]; adev = [10, 10, 20, 30];
tbmax = length(tbmean);
tkmax = length(tkmean);
amax = length(amean);
N = 500;   % adatpontok szama
datav = zeros(3,N);

% Tudas bazis,  ij = 1,2,3 tk, tb fuzzy halmaz indexek, tartalom a fh index
TBN=9;
TB = [3, 1, 1;
      2, 1, 1;
      1, 1, 1;
      1, 2, 4;
      1, 3, 4;
      2, 3, 4;
      3, 3, 4;
      2, 2, 4;
      3, 2, 4];

k=1;
while k<N+1,
    irule = fix(TBN*rand+1);
    i = TB(irule,1); j = TB(irule,2);
           if i==1, v1 = rndgen('L',tbmean(i),tbdev(i));
           elseif i==tbmax, v1 = rndgen('R',tbmean(i),tbdev(i));
           else v1 = rndgen('M',tbmean(i),tbdev(i)); end;
           if j==1, v2 = rndgen('L',tkmean(j),tkdev(j));
           elseif j==tkmax, v2 = rndgen('R',tkmean(j),tkdev(j));
           else v2 = rndgen('M',tkmean(j),tkdev(j)); end;
           v3index = TB(irule,3);
           if v3index==1, v3 = rndgen('L',amean(v3index),adev(v3index));
           elseif v3index==amax, v3 = rndgen('R',amean(v3index),adev(v3index));
           else v3 = rndgen('M',amean(v3index),adev(v3index)); end;

           if (tbmean(1)<=v1)&(v1<=tbmean(tbmax)),
               if (tkmean(1)<=v2)&(v2<=tkmean(tkmax)),
                   if (amean(1)<=v3)&(v3<=amean(amax)),
                       datav(:,k) = [v1, v2, v3]; k = k+1, end; end; end;
end;

figure(1), subplot(3,1,1), plot(datav(1,:)); title('Belso'); grid;
subplot(3,1,2), plot(datav(2,:)); title('Kulso'); grid;
subplot(3,1,3), plot(datav(3,:)); title('Ablak'); grid;

save birlabdata;

datav = zeros(3,N);

k=1;
while k<N+1,
    irule = fix(TBN*rand+1);
    i = TB(irule,1); j = TB(irule,2);
           if i==1, v1 = rndgen('L',tbmean(i),tbdev(i));
           elseif i==tbmax, v1 = rndgen('R',tbmean(i),tbdev(i));
           else v1 = rndgen('M',tbmean(i),tbdev(i)); end;
           if j==1, v2 = rndgen('L',tkmean(j),tkdev(j));
```

```
            elseif j==tkmax, v2 = rndgen('R',tkmean(j),tkdev(j));
            else v2 = rndgen('M',tkmean(j),tkdev(j)); end;
            v3index = TB(irule,3);
            if v3index==1, v3 = rndgen('L',amean(v3index),adev(v3index));
            elseif v3index==amax, v3 = rndgen('R',amean(v3index),adev(v3index));
            else v3 = rndgen('M',amean(v3index),adev(v3index)); end;

            if (tbmean(1)<=v1)&(v1<=tbmean(tbmax)),
                if (tkmean(1)<=v2)&(v2<=tkmean(tkmax)),
                    if (amean(1)<=v3)&(v3<=amean(amax)),
                        datav(:,k) = [v1, v2, v3]; k = k+1, end; end; end;
end;

figure(2), subplot(3,1,1), plot(datav(1,:)); title('Belso'); grid;
subplot(3,1,2), plot(datav(2,:)); title('Kulso'); grid;
subplot(3,1,3), plot(datav(3,:)); title('Ablak'); grid;

save birlabtest;
```

birlabplus.m

```
clear all; close all; rand('seed',0);

tbmean = [18, 21, 24]; tbdev = [1.5, 1.5, 1.5];
tkmean = [5, 20, 35]; tkdev = [6, 6, 6];
amean = [0, 30, 50, 100]; adev = [10, 10, 20, 30];
tbmax = length(tbmean);
tkmax = length(tkmean);
amax = length(amean);
N = 500;   % adatpontok szama
datav = zeros(3,N);

% Tudas bazis,  ij = 1,2,3 tk, tb fuzzy halmaz indexek, tartalom a fh index
TBN=9;
TB = [3, 1, 1;
      2, 1, 1;
      1, 1, 1;
      1, 2, 4;
      1, 3, 4;
      2, 3, 4;
      3, 3, 4;
      2, 2, 4;
      3, 2, 4];

k=1;
while k<N+1,
    irule = fix(TBN*rand+1);
    i = TB(irule,1); j = TB(irule,2);
            if i==1, v1 = rndgen('L',tbmean(i),tbdev(i));
            elseif i==tbmax, v1 = rndgen('R',tbmean(i),tbdev(i));
            else v1 = rndgen('M',tbmean(i),tbdev(i)); end;
            if j==1, v2 = rndgen('L',tkmean(j),tkdev(j));
            elseif j==tkmax, v2 = rndgen('R',tkmean(j),tkdev(j));
            else v2 = rndgen('M',tkmean(j),tkdev(j)); end;
            v3index = TB(irule,3);
            if v3index==1, v3 = rndgen('L',amean(v3index),adev(v3index));
            elseif v3index==amax, v3 = rndgen('R',amean(v3index),adev(v3index));
            else v3 = rndgen('M',amean(v3index),adev(v3index)); end;

            if (tbmean(1)<=v1)&(v1<=tbmean(tbmax)),
                if (tkmean(1)<=v2)&(v2<=tkmean(tkmax)),
```

```
                    if (amean(1)<=v3)&(v3<=amean(amax)),
                        datav(:,k) = [v1, v2, v3]; k = k+1, end; end; end;
end;

figure(1), subplot(3,1,1), plot(datav(1,:)); title('Belso'); grid;
subplot(3,1,2), plot(datav(2,:)); title('Kulso'); grid;
subplot(3,1,3), plot(datav(3,:)); title('Ablak'); grid;

save birlabdata;

datav = zeros(3,N);

TBN=9;
TBtest1 = [3, 1, 1;
      2, 1, 1;
      1, 1, 1;
      1, 2, 4;
      1, 3, 4;
      2, 3, 4;
      3, 3, 4;
      2, 2, 4;
      3, 2, 4];

N1=N/2;

k=1;
while k<N1+1,
    irule = fix(TBN*rand+1);
    i = TBtest1(irule,1); j = TBtest1(irule,2);
           if i==1, v1 = rndgen('L',tbmean(i),tbdev(i));
           elseif i==tbmax, v1 = rndgen('R',tbmean(i),tbdev(i));
           else v1 = rndgen('M',tbmean(i),tbdev(i)); end;
           if j==1, v2 = rndgen('L',tkmean(j),tkdev(j));
           elseif j==tkmax, v2 = rndgen('R',tkmean(j),tkdev(j));
           else v2 = rndgen('M',tkmean(j),tkdev(j)); end;
           v3index = TBtest1(irule,3);
           if v3index==1, v3 = rndgen('L',amean(v3index),adev(v3index));
           elseif v3index==amax, v3 = rndgen('R',amean(v3index),adev(v3index));
           else v3 = rndgen('M',amean(v3index),adev(v3index)); end;

           if (tbmean(1)<=v1)&(v1<=tbmean(tbmax)),
               if (tkmean(1)<=v2)&(v2<=tkmean(tkmax)),
                   if (amean(1)<=v3)&(v3<=amean(amax)),
                       datav(:,k) = [v1, v2, v3]; k = k+1, end; end; end;
end;


TBN=9;
TBtest2 = [3, 1, 1;
      2, 1, 1;
      1, 1, 1;
      1, 2, 2;
      1, 3, 2;
      2, 3, 2;
      3, 3, 2;
      2, 2, 2;
      3, 2, 2];

k=N1+1;
while k<N+1,
    irule = fix(TBN*rand+1);
    i = TBtest2(irule,1); j = TBtest2(irule,2);
           if i==1, v1 = rndgen('L',tbmean(i),tbdev(i));
           elseif i==tbmax, v1 = rndgen('R',tbmean(i),tbdev(i));
```

```
            else v1 = rndgen('M',tbmean(i),tbdev(i)); end;
            if j==1, v2 = rndgen('L',tkmean(j),tkdev(j));
            elseif j==tkmax, v2 = rndgen('R',tkmean(j),tkdev(j));
            else v2 = rndgen('M',tkmean(j),tkdev(j)); end;
            v3index = TBtest2(irule,3);
            if v3index==1, v3 = rndgen('L',amean(v3index),adev(v3index));
            elseif v3index==amax, v3 = rndgen('R',amean(v3index),adev(v3index));
            else v3 = rndgen('M',amean(v3index),adev(v3index)); end;

            if (tbmean(1)<=v1)&(v1<=tbmean(tbmax)),
                if (tkmean(1)<=v2)&(v2<=tkmean(tkmax)),
                    if (amean(1)<=v3)&(v3<=amean(amax)),
                        datav(:,k) = [v1, v2, v3]; k = k+1, end; end; end;
end;

figure(2), subplot(3,1,1), plot(datav(1,:)); title('Belso'); grid;
subplot(3,1,2), plot(datav(2,:)); title('Kulso'); grid;
subplot(3,1,3), plot(datav(3,:)); title('Ablak'); grid;

save birlabtest;
```