6.6.- PRELUCRAREA INREGISTRARILOR

6.6.1- PRELUCRAREA INITIALA

Cele 22 de inregistrari evidentiate in finalul cap. 6.5.c le mutam (numai partile "c") intr-un tabel "Tab" in EXCEL (numai coloanele "a $_z$" si "F $_z$"); apoi le-am transferat in MATLAB, in care recompunem tabelul "Tab", redenumit "t44".

Calculam 12 *indici de monitorizare*:

$X_1 = v$ = viteza de aschiere;

$X_2 = t$ = adancimea de aschiere;

$X_3 = s$ = avansul longitudinal;

$X_4 = \overline{F}_z$ = valoarea medie a fortei principale de aschiere;

X_5 = banda de variatie a lui F_z (inregistrarea continand 960 de esantioane, am impartit-o in 4 parti egale – de cate 240 esantioane – si am calculat maximul si minimul pt. fiecare parte; X_5 este diferenta dintre media maximelor si media minimelor);

X_6 = numarul intersectiilor oscilogramei F_z cu valoarea sa medie $\overline{F}_z$;

X_7 = media densitatii spectrale de putere a lui F_z in banda de frecvente *1 - 2400* Hz ;

X_8 = media densitatii spectrale de putere a lui F_z in banda de frecvente *2401 - 4800* Hz ;

X_9 = media densitatii spectrale de putere a lui F_z in banda de frecvente *4801 - 9600* Hz ;

X_{10} = media densitatii spectrale de putere a lui a_z^{inr} in banda de frecvente *1 - 2400* Hz ;

X_{11} = media densitatii spectrale de putere a lui a_z^{inr} in banda de frecvente *2401 - 4800* Hz ;

X_{12} = media densitatii spectrale de putere a lui a_z^{inr} in banda de frecvente *4801 - 9600* Hz .

Acesti indici, calculati pt. cele 22 de inregistrari, formeaza matricea "x", cu 22 linii si 12 coloane:

Tabelul 6.6.1

x =

Columns 1 through 6

8.6350e+001	5.0000e-001	3.0200e-001	3.8853e+001	8.0770e+000	3.3500e+002
8.6350e+001	5.0000e-001	2.5000e-001	2.6718e+001	7.1227e+000	4.0300e+002
8.4000e+001	5.0000e-001	2.0800e-001	2.0495e+001	5.6573e+000	3.6900e+002
8.4800e+001	5.0000e-001	2.5000e-001	2.9545e+001	6.4752e+000	4.9000e+002
8.4800e+001	5.0000e-001	3.0200e-001	3.1602e+001	6.6115e+000	4.6200e+002
1.3300e+002	1.0000e+000	3.5300e-001	5.9944e+001	5.8277e+000	2.6100e+002
1.3300e+002	1.0000e+000	3.3400e-001	6.0192e+001	5.2483e+000	3.7300e+002
1.3300e+002	1.0000e+000	3.0200e-001	5.8123e+001	6.8160e+000	2.8000e+002
1.3300e+002	1.0000e+000	3.7500e-001	6.3598e+001	5.2313e+001	2.3000e+001
8.3200e+001	1.0000e+000	3.0200e-001	7.5374e+001	6.9523e+000	4.2600e+002
8.3200e+001	1.0000e+000	2.1200e-001	5.7096e+001	6.4070e+000	3.5500e+002
1.3060e+002	1.2000e+000	3.0200e-001	7.7719e+001	9.1675e+000	2.2300e+002
1.3060e+002	1.2000e+000	3.3400e-001	8.1786e+001	7.6339e+000	2.7400e+002
1.3060e+002	1.2000e+000	3.5300e-001	8.9496e+001	7.7021e+000	2.5700e+002
1.3060e+002	1.2000e+000	3.7500e-001	9.1637e+001	7.4976e+000	3.0600e+002
7.9300e+001	1.4000e+000	2.5000e-001	6.6836e+001	8.0429e+000	3.1900e+002
7.9300e+001	1.4000e+000	3.0200e-001	7.6972e+001	9.7469e+000	1.7500e+002
1.2680e+002	1.4000e+000	3.7500e-001	9.0849e+001	2.9854e+001	9.4000e+001
7.7300e+001	1.5000e+000	3.0200e-001	8.8157e+001	8.2474e+000	2.6700e+002
7.5400e+001	2.0000e+000	3.0200e-001	1.0288e+002	9.2016e+000	2.8800e+002
7.5400e+001	2.0000e+000	2.5000e-001	1.0079e+002	9.1334e+000	2.2000e+002
7.1000e+001	3.0000e+000	4.1600e-001	1.5647e+002	1.9494e+001	6.5000e+001

Columns 7 through 12

1.0015e+004	1.3224e+000	1.0440e+000	1.6103e-006	4.6126e-006	2.4307e-005
4.7806e+003	1.6893e+000	9.2422e-001	2.0110e-006	2.5255e-006	1.5556e-005
2.8031e+003	1.3177e+000	6.2598e-001	5.1053e-007	4.9767e-006	1.4966e-005
5.8249e+003	1.0303e+000	1.6544e+000	5.9237e-007	5.3176e-006	1.5565e-005

6.6682e+003 1.0552e+000 1.5393e+000 4.4930e-007 3.5887e-006 1.8399e-005
2.4241e+004 7.4094e-001 6.3477e-001 1.2092e-005 2.9389e-006 1.8129e-005
2.4126e+004 7.9150e-001 7.6272e-001 1.2318e-005 4.2582e-006 1.9059e-005
2.2393e+004 9.0462e-001 5.9583e-001 1.2425e-005 2.8956e-006 2.5164e-005
2.8037e+004 9.8662e-001 9.8148e-001 1.3472e-005 4.6359e-006 2.0672e-005
3.8184e+004 1.3660e+000 1.6390e+000 1.4274e-005 7.1711e-006 1.7458e-005
2.1723e+004 1.0602e+000 6.8397e-001 1.5730e-005 3.3052e-006 1.1485e-005
4.0237e+004 1.2860e+000 6.8552e-001 2.3626e-005 9.1872e-006 1.0742e-005
4.4759e+004 1.1099e+000 4.5915e-001 2.2549e-005 6.2768e-006 7.3750e-006
5.3456e+004 1.0607e+000 8.0039e-001 2.2212e-005 6.4283e-006 8.7312e-006
5.6160e+004 1.2366e+000 1.0382e+000 2.2164e-005 5.6076e-006 8.3173e-006
2.9887e+004 8.8511e-001 6.8384e-001 3.4395e-006 4.6723e-006 1.0022e-005
3.9564e+004 1.3431e+000 6.5909e-001 3.2688e-006 3.7426e-006 1.5528e-005
5.5562e+004 1.1869e+000 7.7111e+000 3.8126e-006 7.5537e-006 4.4260e-003
5.1683e+004 1.2146e+000 6.6422e-001 4.7469e-006 6.9971e-006 8.2586e-006
7.0257e+004 1.7075e+000 1.5874e+000 2.2816e-006 5.8361e-006 1.6512e-005
6.7549e+004 1.6576e+000 6.8277e-001 2.0409e-006 6.8975e-006 1.5319e-005
1.6128e+005 1.5042e+000 1.0078e+000 5.1467e-006 6.5868e-006 1.6759e-005

RNA va recunoaste clasele mai usor, daca ele sunt separate (nu se intrepatrund). Pentru a avea o imagine despre cum sunt dispuse clasele, calculam cateva distante dintre vectorii transpusi pe liniile matricei "x".

Distanta dintre doi vectori *A* si *B*: $|A-B| = \sqrt{(A-B)^T (A-B)}$.

Verificare: $A = [\, x_1 \;\; y_1 \,]^T$; $B = [\, x_2 \;\; y_2 \,]^T$;

$$|A-B|^2 = [\, x_1 - x_2 \;\; y_1 - y_2 \,] \cdot \begin{bmatrix} x_1 - x_2 \\ y_1 - y_2 \end{bmatrix} = (x_1 - x_2)^2 + (y_1 - y_2)^2 ,$$

regasindu-se distanta dintre doua puncte $A = (x_1 , y_1)$ si $B = (x_2 , y_2)$ din planul xOy.

Fiecare vector (*forma*) apares ca un punct in *spatiul formelor* cu 12 dimensions, punctele apartinand fiecarei clase formand un "nor". Am calculat distantele dintre un numar de puncte:

d90_115 = 2.0633e+004; d90_91 = 1.7355e+003; d114_115 = 4.5223e+003;
d91_114 = 1.7843e+004; d89_93 = 3.8041e+003; d116_117 = 2.7049e+003;
d89_116 = 2.9215e+004; d93_117 = 2.8125e+004; d57_84 = 1.0479e+003;
d57_105 = 1.6943e+004; d57_124 = 2.5106e+004; d57_171 = 6.2768e+004;
d54_87 = 3.3490e+003; d127_149 = 1.2119e+004; d87_100 = 3.1516e+004;
d87_149 = 4.5015e+004; d87_168 = 6.3589e+004; d57_80 = 1.9778e+003.

Mai jos, notatia "0C" inseamna ca cele doua puncte apartin aceleiasi clase; "1C" inseamna ca cele doua puncte apartin de doua clase vecine; "2C" - cele doua puncte apartin de doua clase, avand alta clasa intre ele; "3C" - cele doua puncte au alte doua clase intre ele.

Din domeniile de variatie ale distantelor: 0C $\rightarrow$ 1048 ÷ 12 129 ; 1C $\rightarrow$ 16 943 ÷ 31 516 ;

2C $\rightarrow$ 25 106 ÷ 45 015 ; 3C $\rightarrow$ 62 768 ÷ 63 589 , se observa o singura intrepatrundere (in cazul acestui lot limitat de vectori). Deci, putem sa spunem ca *indicii de monitorizare* au fost alesi corect.

Din tabelul 6.6.1 observam in coloana indicelui X_6 ca numai 3 valori sunt de ordinul zecilor (printre care si cea din linia 18, corespunzand inregistrarii nr. 131, din clasa "c_6" (Trepidatii)), restul fiind de ordinul sutelor. Concluzionam ca si celelalte doua inregistrari (linia 9 $\rightarrow$ Inr. 093 si linia 22 $\rightarrow$ Inr. 179) apartin de c_6 , in loc de c_2 , respectiv c_4 . Acest fapt este confirmat in coloana X_5 , valoarea acestui indice pt. cele 3 inregistrari fiind de ordinul zecilor, pt. rest fiind de ordinul unitatilor.

Cu inregistrarile din cap. 6c formam doua loturi, numite **Instruire** si **Clasificare** (conform cap. 6.6.1), primul lot avand 60% din nr. total de inregistrari (cele notate cu "a", "c" si "e"), iar lotul al doilea contine inregistrarile notate cu "b" si "d".

Coloanele "a_z" si "F_z" apartinand inregistrarilor din lotul "Instruire" le mutam intr-un tabel numit "Tabel_a (b, c, . . .)" pe care-l transferam in MATLAB, unde il recompunem in 7 tabele, corespunzatoare celor 7 clase: Tab_C1, . . . ,Tab_C7.

Pe baza valorilor indicilor de monitorizare X_6 si X_5 am mai mutat in clasa c_6 inregistrarile 72 ÷ 75 (din clasa c_1) si 180 (din c_5).

Analiza vizuala a tabelelor "Tabel_a.xls", "Tabel_b.xls", . . . ,releva – cu exceptia a doua cazuri – ca inregistrarile dintr-o clasa au valorile medii ale lui F_z apropiate si in crestere de la o clasa la alta (cu cresterea uzurii), cum era de asteptat.

Prima exceptie o face inregistrarea *092*, la care forta $F_{z\ med}$ este de 5-6 ori mai mica ca la inregistrarile vecine (ce au regimuri de aschiere apropiate), fapt datorat – credem – unei defectiuni in circuitul marcilor tensometrice. Eliminam si aceasta inregistrare.

A doua exceptie o face inregistrarea *103*, la care partile *c, d* si *e* prezinta un $F_{z\ med}$ la jumatatea valorii din partile *a* si *b* (adica se inregistreaza o "cadere" in ultima parte a graficului $F_{z\ med}\ (t)$). Cauza credem ca este aceeasi ca mai sus. Aceasta inregistrare o pastram, considerand ca reteaua este capabila sa surmonteze aceste erori in datele de intrare.

6.6.2- ANALIZA SPECTRALA A INREGISTRARILOR (FFT Discrete Fourier transform)

FFT(X) realizeaza *discrete Fourier transform* (DFT) a vectorului X. FFT(X,N) realizeaza FFT in N puncte, prelungita cu zero-uri daca X un numar mai mic decat N puncte si trunchiata daca numarul este mai mare.

Pentru un vector x (de lungime N), DFT este un vector X (de lungime N), cu elementele:

$$X(k) = \sum_{n=1}^{N} x(n)*\exp(-j*2*pi*(k-1)*(n-1) / N), \quad 1 <= k <= N.$$

Relatiile dintre DFT si coeficientii Fourier "a" si "b" din:

$$x(n) = a0 + \sum_{k=1}^{N/2} (a(k)*\cos(2*pi*k*t(n) / (N*dt) + b(k)*\sin(2*pi*k*t(n) / (N*dt)))$$

sunt: $a0 = X(1) / N = (\sum_{n=1}^{N} x(n)) / N$, $a(k) = 2*real(X(k+1)) / N$, $b(k) = -2*imag(X(k+1)) / N$,

unde x este un semnal discret, de lungime N, esantionat la timpul t cu intervalul dt. Observatie: t(n) / dt = n.

Pentru a testa programul de dezvoltare in serie Fourier, il aplicam functiei:

x = 3 + 4*sin(2*pi*50*t) + 2*sin(2*pi*100*t) + 5*sin(2*pi*350*t);

Rezultatul se prezinta in fig. 6.6.1, in care se observa ca este corect.

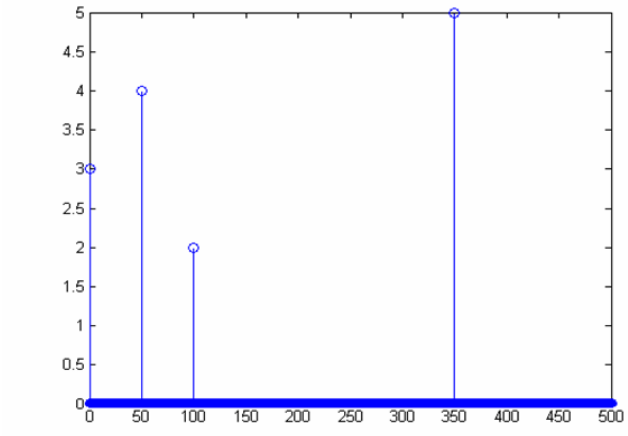

Fig. 6.6.1

Acest program il aplicam in continuare unor inregistrari: 43 (apartinand clasei C1), 89 (C2) ,113 (C3), 164 (C4), 185 (C5) si 64 (C6), pentru analiza spectrala a lui a_z si F_z. Graficele se prezinta – respectiv – in figurile 6.6.2 ÷ 6.6.7.

La inregistrarile din clasa C6 (vezi fig. 6.5.6) se observa ca F_z are o variatie periodica, de perioada mare: $T \approx 100.\Delta t = 100.1/9600$, deci cu frecventa $f = 1 / T = 96$ Hz, care apare si in figura 6.6.7, aceasta fiind **frecventa fundamentala** a vibratiilor

Din reprezentarea grafica $F_z = F_z\ (t)$ pt. inregistrarea nr. 64 se obs. ca amplitudinea vibratiilor este de aproximativ *2.5* N, valoare ce se verifica in fig. 6.6.7.

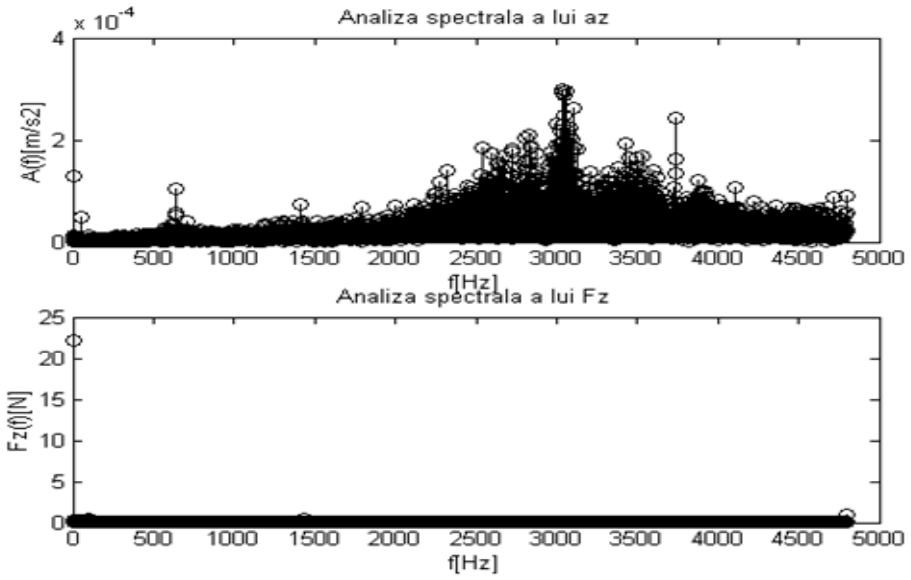

Fig. 6.6.2

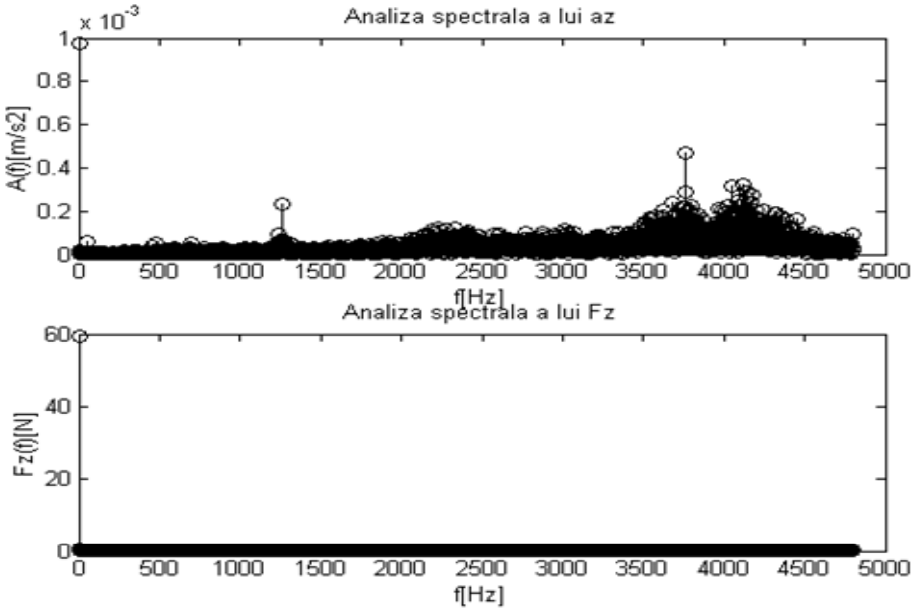

Fig. 6.6.3

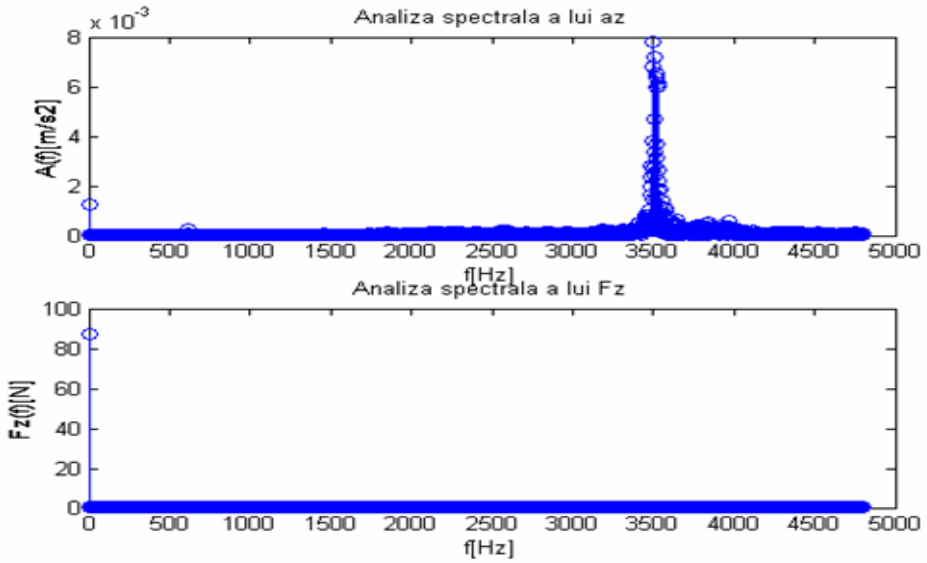

Fig. 6.6.4

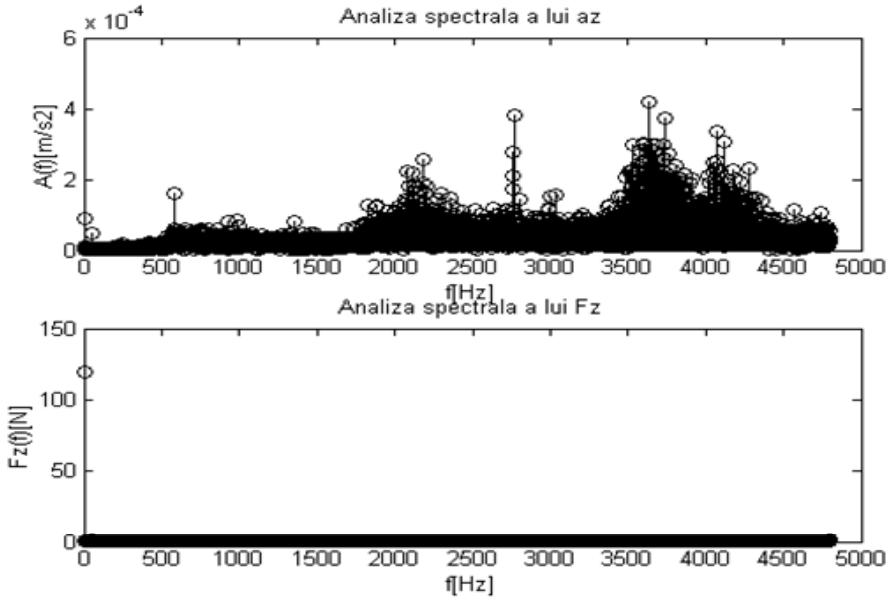

Fig. 6.6.5

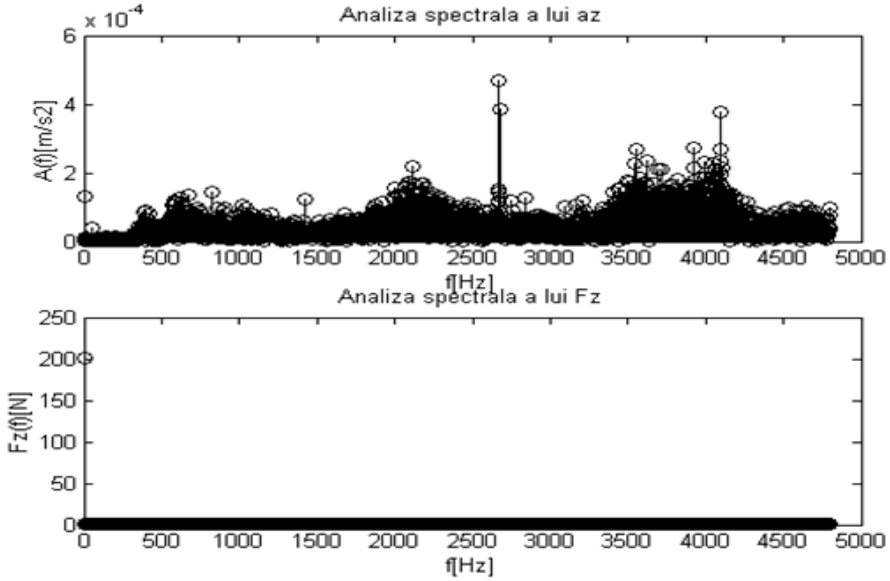

Fig. 6.6.6

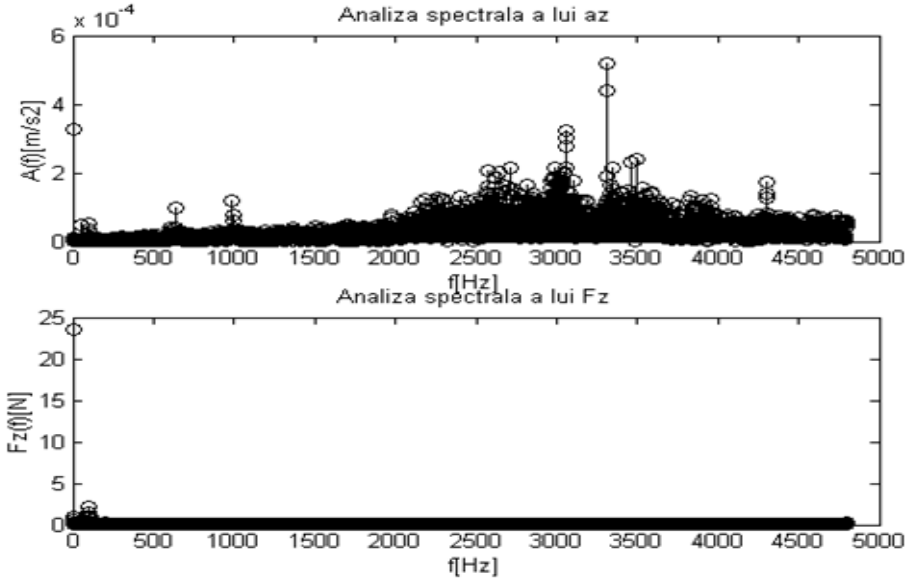

Fig. 6.6.7

6.6.3.- MONITORIZAREA UZURII SCULEI UTILIZAND R.N.A.

A) AMELIORAREA GENERALIZARII

O problema care apare la antrenarea RNA se numeste “potrivire perfecta”: eroarea din setul de antrenare este condusa spre o valoare foarte mica, dar la prezentarea unor noi date retelei, eroarea este mare. Deducem ca reteaua a memorizat exemplele din timpul antrenarii, dar nu poate sa generalizeze in cazuri noi.

O metoda pentru ameliorarea generalizarii RNA consta in utilizarea unor retele suficient de mari. Cu cat reteaua este mai mare, cu atat mai complexe sunt functiile ce le poate crea. O retea mica nu va avea suficienta putere de a potrivi perfect toate datele.

Se observa ca daca numarul parametrilor din RNA este mult mai mic decat numarul datelor din setul de antrenare, atunci probabilitatea *potrivirii perfecte* este foarte mica. Daca se colecteaza mai multe date si creste marimea setului de antrenare, atunci nu sunt necesare metodele de mai jos (ce se aplica in situatiile cand dorim sa realizam maximul, in cadrul unei cantitati limitate de date).

OPRIREA PREMATURA

O alta metoda pentru ameliorarea generalizarii este “oprirea prematura”. In aceasta tehnica datele disponibile sunt impartite in trei subseturi. Primul este setul de antrenare, care este folosit pentru calculul gradientului si pentru actualizarea ponderilor si deplasarilor. Al doilea este setul de validare. Eroarea in acest subset este monitorizata in timpul procesului de antrenare. Eroarea de validare va descreste, in mod normal, in timpul fazei initiale de antrenare, la fel ca si eroarea din setul de antrenare. Dar, cand reteaua incepe sa potriveasca perfect datele, eroarea din setul de validare va incepe – de regula - sa creasca. Cand eroarea de validare creste pe durata unui numar specificat de iteratii, antrenarea este oprita, iar ponderile si deplasarile sunt readuse la minimul erorii de validare.

Eroarea in setul de testare (al 3-lea subset) nu este folosita in timpul antrenarii, ci pentru compararea diferitelor modele. Ea este utila pentru a o plota in timpul procesului de antrenare. Daca aceasta eroare atinge un minim la un numar de iteratii semnificativ diferit decat eroarea din setul de validare, aceasta poate indica o impartire gresita a datelor.

Oprirea prematura poate fi folosita cu oricare dintre functiile de antrenare. Este nevoie doar de trecerea datelor de validare in functia de antrenare. Nu trebuie folosit un algoritm care converge prea repede. Daca folosim un asemenea algoritm (ca *trainlm*), va trebui sa fixam parametrii de antrenare astfel incat convergenta sa fie relativ lenta (de exemplu: setam “mu” la o valoare relativ mare, cum ar fi 1, si setam “mu_dec” si “mu_inc” la valori apropiate de 1, cum ar fi 0.8 si 1.5, respectiv). Functiile de antrenare *trainscg* si *trainrp* sunt recomandate in cazul opririi premature.

Alegerea setului de validare este de asemenea important. El ar trebui sa fie reprezentativ pentru toate elementele din setul de antrenare.

Este indicat sa antrenam reteaua incepand cu diferite conditii initiale. E posibil ca orice metoda sa dea gres in anumite situatii. Prin testarea diferitelor conditii initiale, se poate verifica robustetea retelei.

B) PRE-PROCESAREA SI POST-PROCESAREA

Antrenara retelelor neuronale poate fi mai eficienta daca anumite pre-procesari sunt executate pe intrarile in retea si tintele.

MEDIA SI DEVIATIA STANDARD (prestd, poststd, trastd)

Inainte de antrenare, este util sa scalam intrarile si tintele astfel incat sa se incadreze intr-un domeniu specificat. O abordare in acest sens este de a normaliza media si deviatia standard in setul de antrenare. Aceasta procedura este aplicata cu functia *prestd*. Ea normalizeaza intrarile si tintele astfel incat ele vor avea media zero, iar deviatia standard unitara. Urmatorul cod ilustreaza folosirea lui *prestd*: [pn,meanp,stdp,tn,meant,stdt] = prestd(p,t);

Intrarile in retea si tintele initiale sunt date in matricile *p* si *t*. Intrarile si tintele normalizate *pn* si *tn*, care sunt returnate, vor avea mediile nule si deviatiile standard unitare. Vectorii *meanp* si *stdp*

contin mediile si deviatiile standard ale intrarilor originale, iar vectorii *meant* si *stdt* contin mediile si deviatiile standard ale tintelor originale. Dupa ce reteaua a fost antrenata, acesti vectori ar trebui folositi pentru transformarea viitoarelor intrari aplicate retelei. Ei astfel devin parte din retea, la fel ca ponderile si deplasarile din retea.
Daca *prestd* este folosita pentru a scala intrarile si tintele, atunci iesirea retelei este antrenata sa produca iesiri cu medie nula si deviatie standard unitara. Daca vrem sa convertim aceste iesiri in aceleasi unitati care erau folosite pentru tintele originale, atunci trebui sa folosim rutina *poststd*. In urmatorul cod, vom simula reteaua care a fost antrenata in codul anterior si apoi convertim iesirea retelei inapoi la unitatile originale.

```
an = sim(net,pn);
a = poststd(an,meant,stdt);
```

Iesirea *an* din retea corespunde tintelor normalizate *tn*. Iesirea ne-normalizata *a* este in aceleasi unitati ca si tintele originale *t*.
Daca *prestd* este folosita pentru preprocesarea datelor setului de antrenare, atunci ori de cate ori reteaua antrenata este folosita pentru intrari noi, ele trebuie pre-procesate cu mediile si derivatiile standard care au fost calculate pentru setul de antrenare. Aceasta se poate face cu rutina *trasd*. In urmatorul cod aplicam un nou set de intrari retelei pe care tocmai am antrenat-o.

```
pnewn = trastd(pnew,meanp,stdp);
anewn = sim(net,pnewn);
anew = poststd(anewn,meant,stdt);
```

ANALIZA COMPONENTELOR PRINCIPALE (prepca, trapca)
In unele situatii, dimensiunea vectorului de intrare este mare, dar componentele vectorilor sunt corelati foarte mult. In aceasta situatie este bine sa se reduca dimensiunea lor. O procedura eficace pentru executarea acestei operatii este *analiza componentelor principale*. Aceasta tehnica are trei efecte: ortogonalizeaza componentele vectorilor de intrare (astfel ca ei sa nu fie corelati intre ei); ordoneaza componentele ortogonale rezultate (componente principale) astfel incat cele cu variatiile maxime sunt primele; elimina acele componente care contribuie cel mai putin la variatia din setul de date. Urmatorul cod prezinta folosirea *prepca*, care executa analiza componentelor principale.

```
[pn,meanp,stdp] = prestd(p);
[ptrans,transMat] = prepca(pn,0.02);
```

Se observa ca mai intai se normalizeaza vectorii de intrare folosind *prestd*, astfel incat sa aiba medie zero si varianta unitara. Aceasta este o procedura standard cand folosim componentele principale. In acest exemplu, al doilea argument din *prepca* este 0.02. Aceasta inseamna ca *prepca* elimina acele componente principale care contribuie cu mai putin de 2% la variatia totala din setul de date. Matricea *ptrans* contine vectorii de intrare transformati. Matricea *transMat* contine matricea de transformare a componentelor principale. Dupa ce reteaua a fost antrenata, aceasta matrice trebuie folosita pentru transformarea urmatoarelor intrari aplicate retelei. Ea devine o parte a retelei, asa cum sunt ponderile si deplasarile. Daca multiplicam vectorii de intrare normalizati *pn* cu matricea *transMat*, vom obtine vectorii de intrare transformati *ptrans.*
Daca *prepca* este folosita pentru pre-procesarea datelor din setul de antrenare, atunci ori de cate ori reteaua antrenata este folosita cu intrari noi, ele trebuie pre-procesate cu matricea de transformare care a fost calculata pentru setul de antrenare. Acest lucru se poate efectua cu rutina *trapca*. In urmatorul cod, aplicam un set nou de intrari unei retele pe care am antrenat-o deja.

```
pnewn = trastd(pnew,meanp,stdp);
pnewtrans = trapca(pnewn,transMat);
a = sim(net,pnewtrans);
```

6.6.4- RULAREA R.N.A.
Vom aplica tehnicile descrise mai sus.
Setul de antrenare va contine inregistrarile din lotul “Instruire”, deci 60% din numarul total de inregistrari. Setului de validare ii alocam inregistrarile “b” din lotul “Clasificare”, iar in setul de testare vor intra inregistrarile “d”; fiecare set are deci 20 % din numarul inregistrarilor.

RNA are *3* nivele, cu $s_1 = 23$ neuroni, $s_2 = 27$, $s_3 = 7$ (= numarul claselor). Matricea de intrare *p* are dimensiunile *12* (indici de monitorizare) *x 655* (inregistrari), iar matricea de iesire *y* are dimensiunile *7* (clase) *x 655*. Functiile de antrenare sunt: *tf1 = purelin, tf2 = tansig, tf2 = logsig*, deci vectorii de iesire au *7* elemente, cu valori in domeniul (0, 1).
In secventa cu instructiuni:

[ma, ia]=max(a); [mt, it]=max(t); c = ia = = it; n = 1: 655; nc = [n, c];

mt = 1 (evident), iar "ma" este valoarea maximului pe coloana lui "a"; daca indicii "ia" si "it" (care dau pozitia respectivelor maxime) sunt identici, atunci maximele pe coloanele lui"a" coincid cu maximele pe coloanele lui "t" si in matricea *c (1x655)* elementul respectiv este egal cu *1*, iar in caz contrar ia valoarea *0*. "nc" este o matrice cu doua linii, pe prima linie numarandu-se inregistrarile, iar a doua linie are elementele *1* (corect), sau *0* (eroare).
Primele rulari (cu functiile de antrenare *trainrp, trainscg,* etc) au indicat erori numai in pozitiile corespunzatoare inregistrarilor din clasele *c5* si *c7*, adica la clasele care au cele mai putine inregistrari. Marirea numarului de inregistrari – fara a efectua noi experiente – se poate face adaugand, intr-o clasa aceleasi inregistrari de mai multe ori, eventual afectate de un zgomot de medie *0.1*; de exemplu, daca inregistrarile sunt cuprinse in tabelul "Tabel 112", ce are dimensiunile *960 x 6*, atunci se adauga inregistrarile: Tabel 112 + randn(960,6)*0.1.
Ca urmare:
- inpatrim inregistrarile din clasa *c5*: la primul set atasam un zgomot de medie *0.1*, al doilea set este identic cu cel original, iar la al treilea set atasam un zgomot de medie *0.15*;
- intreim inregistrarile din clasa *c7*, procedand analog ca mai sus (numai primul si al doilea set).

In concluzie, clasele din cap. 6.5.c au forma finala:
- in clasa "c_1" ($VB_{max} \le 0{,}2$ mm) se incadreaza inregistrarile 43 ÷ 63, 77 ÷ 87, adica 32 Inregistrari (cu cate 4800 esantioane);
- clasa "c_2" ($0{,}2 < VB_{max} \le 0{,}4$ mm) $\rightarrow$ 89 ÷ 91, 94, 96 ÷ 112 $\Rightarrow$ 21 Inregistrari;
- "c_3" ($0{,}4 < VB_{max} \le 0{,}6$ mm) $\rightarrow$ 113 ÷ 118, 124 ÷ 130, 132 ÷ 144, 146 ÷ 163 $\Rightarrow$ 44 Inr.;
- "c_4" ($0{,}6 < VB_{max} \le 0{,}7$ mm) $\rightarrow$ 164 ÷ 171, 174 ÷ 177 $\Rightarrow$ 12 Inr.;
- "c_5" ($VB_{max} > 0{,}7$ mm) $\rightarrow$ 182 ÷ 184 $\Rightarrow$ 3 Inr. + 3 Inr. (cu zgomot) + 3 Inr. + 3 Inr. (cu zgomot) = 12 Inr.;
- "c_6" (Trepidatii) $\rightarrow$ 64, 67, 72 ÷ 76, 93, 95, 131, 172, 179, 180, 185 $\Rightarrow$ 14 Inr.;
- "c_7" (Strunjire in gol) $\rightarrow$ 187 ÷ 191 $\Rightarrow$ 5 Inr. + 5 Inr. (cu zgomot) + 5 Inr. $\Rightarrow$ 15 Inr.

Deci, *c5* va contine *60* de inregistrari, *c7 – 75* inregistrari, iar numarul coloanelor din matricele de mai sus creste de la *655* la *750*.
Cu instructiunile: ind = find(c = = 0); dim = length(ind); er = dim / 750,
Gasim – respectiv – indicii elementelor lui "c" care sunt nuli, numarul de elemente din "ind" si eroarea retelei.
Mai jos se prezinta rezultatele unei rulari:
R = 8 ; Q = 750
Exista o redundanta in setul de date, pentru ca *analiza componentelor principale* a redus dimensiunea vectorilor de intrare de la *12* la *8*.

```
TRAINRP, Epoch 0/300, MSE 1.53359/0, Gradient 0.292509/1e-006
TRAINRP, Epoch 25/300, MSE 0.487294/0, Gradient 0.0150939/1e-006
TRAINRP, Epoch 43/300, MSE 0.477332/0, Gradient 0.0148044/1e-006
TRAINRP, Validation stop.
ind =
  Columns 1 through 18
440  441  445  446  447  448  449  450  454  455  456  457  458  459  463  468  472  473
  Columns 19 through 36
474  475  476  477  481  482  483  519  520  521  533  545  594  595  596  646  647  658
  Column 37
672
```

dim = 37 ; er = dim / 750 = 0.0493 = 4.93 %.

Un mijloc de diagnostic util este de a plota erorile de antrenare, validare si testare pentru a verifica progresul antrenarii. Resultatul se prezinta in figura 6.6.8.

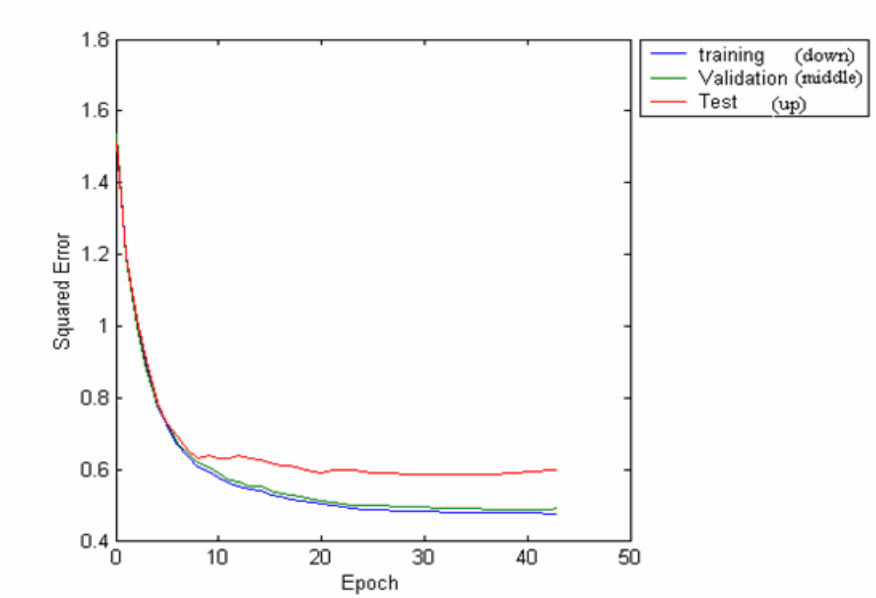

Fig. 6.6.8

Rezultatul este rezonabil, pentru ca erorile de validare si testare au caracteristice similare.
In alte rulari:
- efectuate in aceleasi conditii, rezultatele au fost de doua ori ca mai sus, iar o data au fost:
  51 epoci; dim = 27 ; er = 3.6 %;
- fara functia “init” (de reinitializare a ponderilor si deplasarilor), in doua rulari erorile au fost 4.67% si 13.3%;
- cu functia de antrenare *trainscg* fara “init” eroarea a fost de 5.33%, iar cu “init”- 25.6%.

6.6.5- CALCULE STATISTICE

In matricea de intrare *p*, avand dimensiunile 12 (indici de monitorizare) x 750 (inregistrari), calculam pt. fiecare coloana (inregistrare):
- media aritmetica (a celor 12 indici) μ ;
- abaterea medie patratica (deviatia standard) $\sigma = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N}(x_k - \mu)^2}$, unde $N = 12$;

rezultatele se prezinta in fig. 6.6.9, in care clasei c_1 ii sunt alocate inregistrarile $1 \div 160$, $c_2 \rightarrow 161 \div 265$, $c_3 \rightarrow 266 \div 485$, $c_4 \rightarrow 486 \div 545$, $c_5 \rightarrow 546 \div 605$, $c_6 \rightarrow 606 \div 675$, $c_7 \rightarrow 676 \div 750$.
Se obs. ca cele doua grafice au aceeasi alura, iar clasele sunt usor de separat (in functie de valorile lui μ si σ). Concluzionam ca indicii de monitorizare au fost alesi corect.

6.7.- MONITORIZAREA

Monitorizarea unui strung inseamna ca in timpul procesului **continuu** de aschiere (fig. 6.7.1) sa se conecteze aparatul “Spider”, care sa transmita la PC o inregistrare pe baza careia RNA sa spuna in ce clasa se situeaza prelucrarea (conform tabelului 6.2.2). Apoi “Spider” se conecteaza din nou, s.a.m.d. In cazul in care clasa este mai mare de 3 (deci “anormala”), PC trebuie sa dea un semnal sonor, sau sa opreasca prelucrarea.
Mai detaliat: inregistrarea are 4800 de esantioane, fiind un tabel in EXCEL cu 4800 linii si 3 coloane ($A = \varepsilon_1^{inreg}$, $B = \varepsilon_2^{inreg}$, $C = a_z$). Transferul Spider $\rightarrow$ PC se face in cca. 1 min. Din acest tabel se selecteaza alt tabel (mai mic) cu 960 linii si 4 coloane, primul element fiind ales aleator, la o locatie mai mare de A500. In a IV-a coloana se calculeaza functia F_z (= 1136*A-1136*B), conform relatiei (6.4.12). Acest tabel se transfera in MATLAB, unde se calculeaza cei 12 indici de monitorizare care se prezinta la intrarea RNA, cum se arata la finalul cap. 6.6.4.

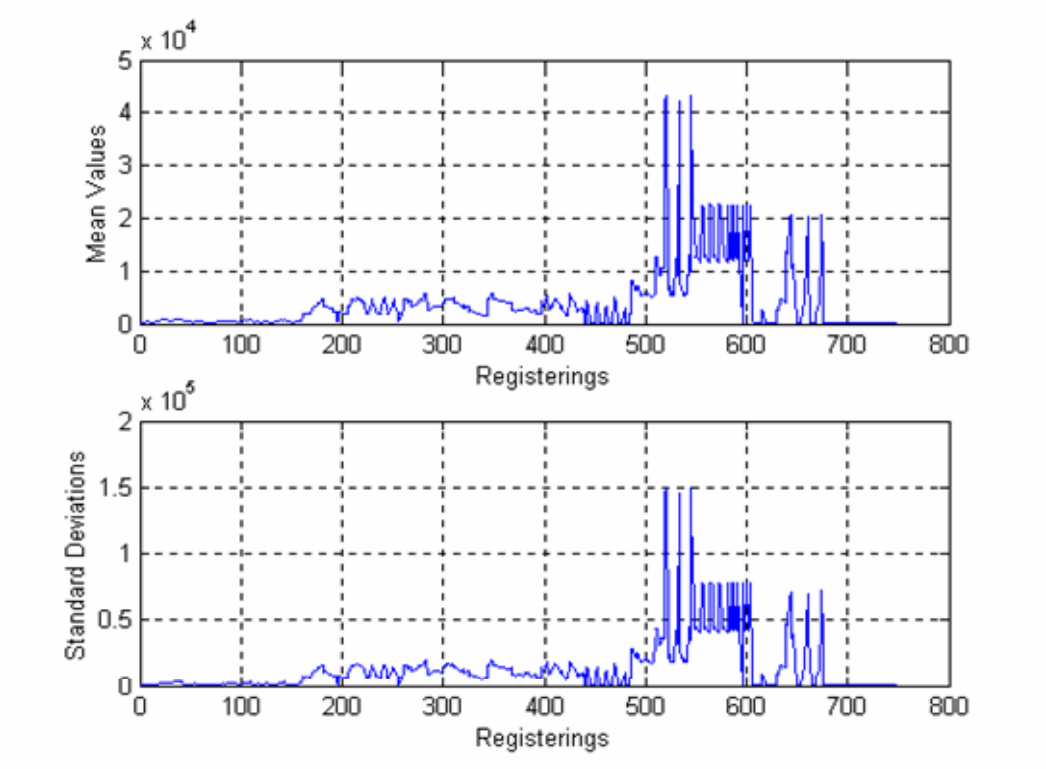

Fig. 6.6.9

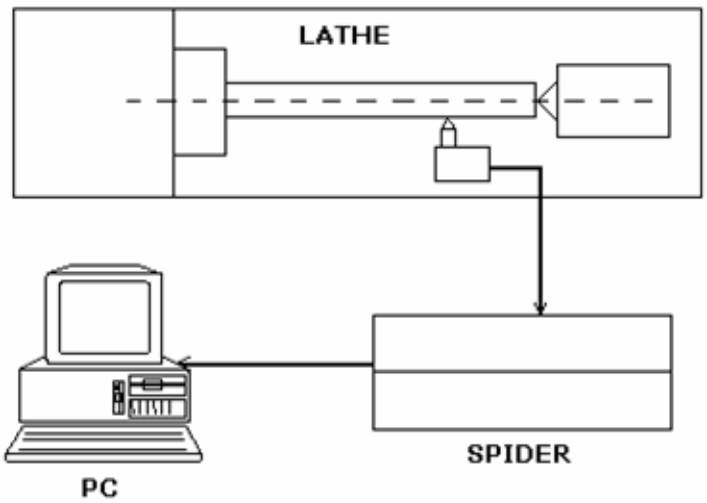

Fig. 6.7.1

Pentru a vedea cum raspunde RNA am utilizat cateva dintre inregistrarile initiale (prezentate in finalul cap. 6.6.1); rezultatele sunt cele din tab. 6.7.1.

Tab. 6.7.1

Nr. crt.	Inreg.	Clasa	v [m/min]	t [mm]	s [mm/rot]	Epoci	Eroarea %	Clasa din RNA	Timpul [min]
0	1	2	3	4	5	6	7	8	9
1	046	1	69	0.5	0.25	43	4.93	1	10
2	183	5	71	3	0.302	40	4.4	5	8
3	168	4	75.4	2	0.302	76	2.5	4	6
4	174	4	120.6	2	0.416	67	3.47	4	8
5	099	2	83.2	1	0.334	43	4.93	2	
6	190	7	0	0	0	47	3.6	7	7
7	095	6	83.2	1	0.353	47	23.87	6	7

La primele 4 inregistrari, dupa prima rulare s-a mai facut si o a II-a rulare, ce a durat 2 min.,

clasa data de RNA coincizand cu cea de la prima rulare.
Coincidenta valorilor din coloanele 2 si 8 ne arata ca RNA furnizeaza rezultate corecte.
In cadrul monitorizarii **reale**, cand aschierea va fi continua, pt. evitarea "necompensarii termice" (vezi cap. 6.4.A) trebuie folosita racirea cu apa a cutitului. Dar, apa poate provoca deranjamente in circuitele marcilor tensometrice, desi acestea sunt protejate (cu Poxipol). Ca urmare, pentru acest stadiu al experimentului va trebui sa eliminam marcile tensometrice si sa masuram componentele fortei de aschiere cu dinamometrul KISLER (Austria).
In plus, coloana nr. 9 din tabelul 6.7.1 ne arata ca hard-ul actual ne furnizeaza informatii tardive, adica aflam ca scula este – de exemplu - in clasa 4 (Uzura severa), cand este posibil ca ea sa fi ajuns in clasa 5 (Deteriorare), sau 6 (Trepidatii). Deci este necesar un PC *specializat*, care sa reduca cat mai mult timpul de raspuns.
Pentru a verifica eficacitatea soft-ului, vom relua rularile ca in tabelul 6.7.1, dar pentru o treime din cele 44 de inregistrari din clasa a 3-a. Rezultatul se prezinta in tab. 6.7.2, in care "L" este
Tab. 6.7.2

Nr. crt.	Inreg.	v [m/min]	t [mm]	s [mm/rot]	L [mm]	Epoci	Eroarea %	Clasa din RNA
0	1	2	3	4	5	6	7	8
1	115	130.6	1.2	0.334	245	40	4.4	**2**
2	118	130.6	1.2	0.416	90	76	2.53	3
3	126	79.3	1.4	0.292	770	43	4.93	3
4	129	79.3	1.4	0.353	630	43	4.93	3
5	133	63.4	1.4	0.353	425	40	4.4	**4**
6	136	63.4	1.4	0.292	300	76	2.53	3
7	139	63.4	1.4	0.212	200	64	2.67	3
8	142	63.4	1.4	0.167	120	67	3.47	3
9	146	77.3	1.5	0.375	780	43	4.93	3
10	149	77.3	1.5	0.302	615	47	3.6	3
11	152	77.3	1.5	0.25	480	47	23.87	3
12	155	123.7	1.5	0.177	340	40	4.4	3
13	158	123.7	1.5	0.146	230	31	15.2	3
14	161	123.7	1.5	0.118	130	76	2.53	3

distanta de la mijlocul zonei aschiate la universal. Se observa ca numai inregistrarile de la nr. 1 si 5 nu au dat clasa corecta (3), ci o clasa vecina. Alte doua rulari realizate pt. fiecare dintre aceste inregistrari – cu modificarea locatiei primului element din tabelul (960 x 4) – au indicat clasa corecta (3). Deci la 25 de rulari, numai doua au fost eronate, eroarea fiind de 8 %. Consideram ca aceasta eroare se va micsora daca vom lua mai multe inregistrari in tab. 6.7.1 si 2.

6.8.- METODA FUZZY C-MEANS [2,6]

Pentru comparatie cu metoda RNA, vom clasifica cele 750 de inregistrari utilizand *metoda fuzzy c - means*.
Clusterizarea fuzzy este o metoda de determinare a clusterilor optimali intr-un spatiu vectorial, pe baza **normelor euclidiene** pentru distanta dintre vectori.
Fuzzy c-means (FCM) este o metoda de clusterizare a datelor in care fiecare punct de date apartine unui cluster, cu un anumit grad de apartenenta. Cu aceasta metoda se pot grupa puncte de date dintr-un spatiu multidimensional intr-un numar de clustere diferite.
Pentru un set de date din tabelul 6.6.1: $X = (\boldsymbol{x}_1 , \boldsymbol{x}_2 , \ldots , \boldsymbol{x}_N)$, (6.8.1)
obiectivul este sa clasificam datele in *n* clusteri fuzzy (clase). O valoare de apartenenta care descrie gradul de apartenenta a fiecarei date la fiecare clasa este:

$$\mu_{k,j} (\boldsymbol{x}_k) \in [0 , 1] , (1 \le j \le n , 1 \le k \le N) . \quad (6.8.2)$$

Matricea de *partitie* este definita astfel: $\mu(n)=\begin{bmatrix} \mu_{11}(x_1) & . & . & \mu_{1N}(x_N) \\ . & & & . \\ . & & & . \\ \mu_{n1}(x_1) & . & . & \mu_{nN}(x_N) \end{bmatrix}$, (6.8.3)

unde valorile de apartenenta sunt supuse la urmatoarele conditii:

$$\sum_{j=1}^{n} \mu_{kj} = 1 \ , \ (k = 1,\ldots,N) \ , \qquad (6.8.4)$$

adica suma valorilor de apartenenta a datelor la toti clusterii fuzzy (clase) trebuie sa fie unitara.
Indicele de performanta pentru matricea de partitie este:

$$J = \sum_{k=1}^{N}\sum_{j=1}^{n} (\mu_{kj})^w \, \| \boldsymbol{x}_k - \boldsymbol{v}_j \|^2 \ , \qquad (6.8.5)$$

unde $\boldsymbol{v}_j$ este un vector pseudo-centru de greutate a clasei *j*:

$$\boldsymbol{v}_j = \frac{\sum_{k=1}^{N} (\mu_{kj})^w \cdot x_k}{\sum_{k=1}^{N} (\mu_{kj})^w}, \ (j = 1, 2, \ldots, n) \qquad (6.8.6)$$

si $w > 1$ este o pondere a valorilor de apartenenta; $\| \boldsymbol{x}_k - \boldsymbol{v}_j \|$ este norma euclidiana, adica distanta dintre $\boldsymbol{x}_k$ si $\boldsymbol{v}_j$: $\| \boldsymbol{x}_k - \boldsymbol{v}_j \| = \sqrt{\sum_{i=1}^{m} (x(k,i) - v(j,i))^2}$. (6.8.7)

Matricea de partitie *optima* se obtine cand indicele de performanta *J* isi atinge minimul, adica atunci cand suma fluctuatiilor interioare claselor are valoarea minima.
In faza de instruire, procedura este:
1.- Construirea matricei de partitie. Euristic se determina $\mu^t{}_{k,j}$, care satisface (6.8.4):

$$\mu(n)=\begin{bmatrix} \mu^t_{11}(x_1) & . & . & \mu^t_{1N}(x_N) \\ . & & & . \\ . & & & . \\ \mu^t_{n1}(x_1) & . & . & \mu^t_{nN}(x_N) \end{bmatrix} , \qquad (6.8.8)$$

Sari la pasul 2, luand $t = 0$.
2.- Calculeaza vectorul cu pseudo-centrele de greutate pentru fiecare cluster din matricea de partitie:

$$\boldsymbol{v}_j^t = \frac{\sum_{k=1}^{N} (\mu^t_{kj})^w \cdot x_k}{\sum_{k=1}^{N} (\mu^t_{kj})^w}, \ (j = 1, \ldots, n) . \qquad (6.8.9)$$

3.- Modifica fiecare valoare de apartenenta:

$$\mu^{t+1}_{kj}(x_k) = \frac{1}{\sum_{\alpha=1}^{n} \left(\frac{\| x_k - v^t_j \|}{\| x_k - v^t_\alpha \|} \right)^{1/(w-1)}} \ , \text{ if } \ \boldsymbol{x}_k \neq \boldsymbol{v}_j^t \ ; \qquad (6.8.10)$$

pana cand $\boldsymbol{x}_k = \boldsymbol{v}_j^t$, apoi: $\mu^{t+1}{}_{k,j}(\boldsymbol{x}_k) = 1$, daca $j = \alpha$; $\mu^{t+1}{}_{k,j}(\boldsymbol{x}_k) = 0$, daca $j \neq \alpha$.
Matricea de partitie modificata este re-organizata:

$$\mu^{t+1}(n)=\begin{bmatrix} \mu^{t+1}_{11}(x_1) & . & . & \mu^{t+1}_{1N}(x_N) \\ . & & & . \\ . & & & . \\ \mu^{t+1}_{n1}(x_1) & . & . & \mu^{t+1}_{nN}(x_N) \end{bmatrix} , \qquad (6.8.11)$$

4.- Se verifica daca convergenta matricei de partitie este indeplinita:

$$\left| \mu^{t+1}_{kj} - \mu^{t}_{kj} \right| < \varepsilon, \qquad (6.8.12)$$

unde ε este un prag de convergenta. Daca convergenta nu este indeplinita, sari la pasul 2 luand $t = t + 1$.

In faza de clasificare, pentru o *noua* inregistrare $\boldsymbol{x}$, avand centrele clusterilor [expresia finala din 6.8.9)], se calculeaza gradul fuzzy a noii inregistrari cu o relatie de tipul (6.8.10):

$$\mu_{kj}(x) = \frac{1}{\sum_{\alpha=1}^{n} \left(\frac{\| x - v_j \|}{\| x - v_\alpha \|} \right)^{1/(w-1)}} \quad , \text{ if } \quad \boldsymbol{x} \neq \boldsymbol{v}_j \ ; \qquad (6.8.13)$$

pana cand $\boldsymbol{x} = \boldsymbol{v}_j$, apoi: $\mu_j (\boldsymbol{x}) = 1$, daca $j = \alpha$ si $\mu_j (\boldsymbol{x}) = 0$, daca $j \neq \alpha$.

In concluzie, conditiile de proces se identifica prin gradele fuzzy maxime.

Functia *fcm* din toolbox-ul “Fuzzy Logic” incepe cu o alegere initiala a centrelor clusterilor, in locatia medie a fiecarui cluster. Aceasta alegere este – cel mai adesea - incorecta. In plus, *fcm* atribuie fiecarui punct de date un grad de apartenenta la fiecare cluster. Prin actualizarea iterativa e centrelor clusterilor si a gradelor de apartenenta pentru fiecare punct de date, *fcm* muta iterativ centrele clusterilor spre locatia "corecta" din interiorul unui set de date. Aceasta iteratie se bazeaza pe minimizare unei functii obiectiv care reprezinta distanta de la oricare punct de date la centrul clusterului, ponderata cu gradul de apartenenta al punctului de date.

fcm este o functie ce are la iesire o lista de centre de clusteri si cateva grade de apartenenta pentru fiecare punct de date.

Descriere: *[center, U, obj_fcn] = fcm(data, cluster_n)* aplica metoda clusterizarii fuzzy c-means unui set de date.

Argumentele de intrare in aceasta functie sunt: *data* (setul de date de clusterizat; fiecare linie este un punct de date inregistrate); *cluster_n* (numarul clusterilor (> 1)).

Argumentele de iesire din aceasta functie sunt: *center* (centrele finale ale clusterilor, in care fiecare linie este un centru); *U* (forma finala a matricei de partitie fuzzy (sau matricea functiilor de apartenenta)); *obj_fcn* (valorile functiilor obiectiv in timpul iteratiilor).

fcm(data,cluster_n,options) are un argument aditional *options*, pentru a controla parametrii clusterizarii, a introduce un criteriu de oprire si/sau a seta iteratiile:

- options(1): exponent for the partition 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);

Daca nu se specifica nici-o optiune, se utilizeaza valorile implicite. Procesul clusterizarii se opreste cand se atinge numarul maxim de iteratii, sau cand castigul la functia obiectiv dupa doua iteratii consecutive este mai mic decat minimul castigului specificat.

Pentru argumentul de intrare “data” egal cu matricea p’= x (750x12) din cap. 6.6.5 (pe o coloana din U (7x750) fiind gradele de apartenenta ale unui vector la cele 7 clase), programul ce atribuie vectorul clasei pentru care gradul de apartenenta este maxim, da rezultatul din fig. 6.8.1 si mai jos.

Iteration count = 1, obj. fcn = 647350640000.421870
. . .
Iteration count = 31, obj. fcn = 121863345485.359880
. . .
Iteration count = 61, obj. fcn = 120594760756.125920
. . .
Iteration count = 91, obj. fcn = 120594676310.747500
. . .
Iteration count = 108, obj. fcn = 120594676310.453870
Iteration count = 109, obj. fcn = 120594676310.453890
Iteration count = 110, obj. fcn = 120594676310.453890
Coloana nr. 100 din U: U100 = [0.0006 0.9990 0.0000 0.0001 0.0002 0.0000 0.0000] ^{T}

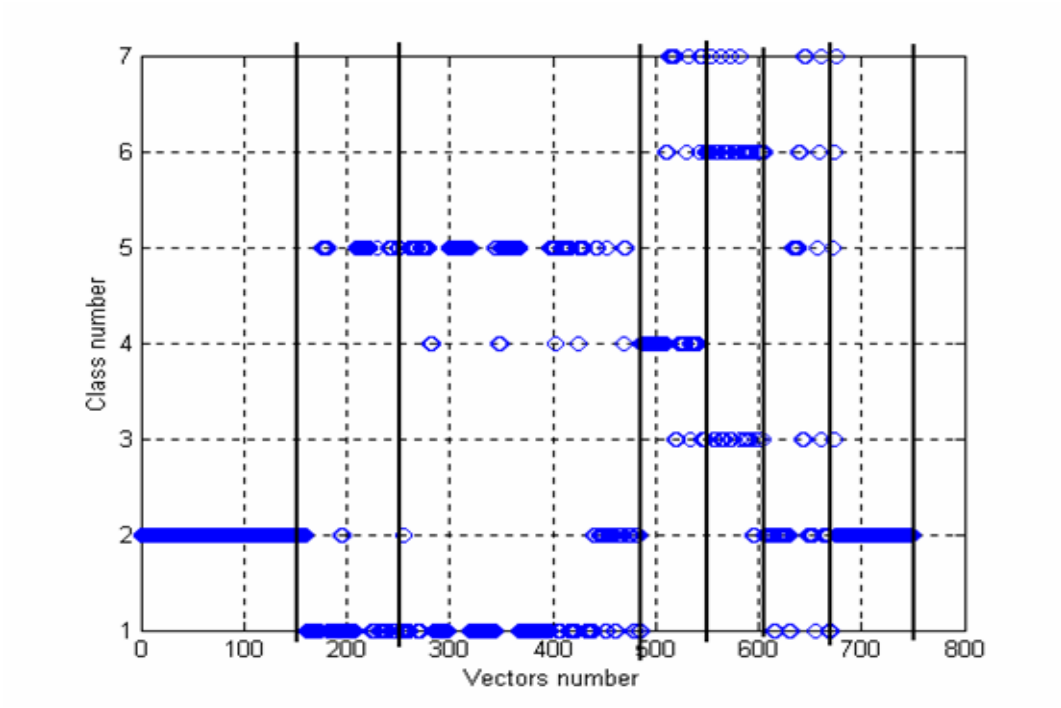

Fig. 6.8.1

O a doua rulare are ca rezultat o figura analoaga cu fig. 6.8.1, orizontalele avand aceeasi componenta, dar pe alte nivele.

Un program analog, in care s-au utilizat numai 4 coloane din matricea x (750x12) si anume, cele mai importante: 4 (F_z), 6 (Nr. intersectiilor), 7 (Densitatea spectrala F_z) si 10 (Densitatea spectrala a_z), furnizeaza aceleasi rezultate.

De asemenea, acelasi rezultat s-a obtinut si cand setul de date pentru clusterizat a avut 2 coloane: marimile din fig. 6.6.9.

Interpretarea figurii 6.8.1 este urmatoarea: clasele stabilite la metoda RNA (reamintite langa fig. 6.6.9) se regasesc partial in fig. 6.8.1, fiind numerotate intr-o alta ordine (c_1 este acum clasa nr. 2, dar c_7 este tot 2, ceea ce nu este corect; c_2 este 1, majoritatea inregistrarilor fiind pe nivelul 1; c_3 este impartita pe nivelele 1 si 5, ceea ce nu convine; c_4 este 4, majoritatea inregistrarilor fiind pe nivelul 4, etc.). Se obs. ca sunt alocate putine inregistrarilor la 3 clase (3, 4 si 7). In concluzie, metoda fuzzy c - means da erori mult mai mari in comparatie cu metoda RNA.

BIBLIOGRAFIE

1- Balan, G, 2002, The monitoring of a lathe using an artificial neural network, Grant type A nr. 33 445, Theme 19, Cod CNCSIS 451

2- Bezdek, J.C., 1981, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York

3- Du, R., Elbestawi, M. A., Wu, S. M., 1995, Automated Monitoring of Manufacturing Processes, Part 1: Monitoring Methods, Part 2: Applications, ASME Journal of Engineering for Industry, may, vol. 117, Part 1-pp. 121 - 132, Part 2 – pp .133 - 141.

4- Epureanu, A., 1983, Technologies in machine building, EDP, Romania.

5- N. Oancea, s.a. *Procese de aschiere. Experimente de laborator*, Ed. TEHNICA-INFO Chisinau, 2002

6- Ryan, J., Ryan, M., Power, J., 1994, Using fuzzy logic, Prentice Hall

7- STAS 12046 / 1 - 81**,** Cutting life testing. Wear. General notions

8- STAS 12046 / 2 - 81 , Cutting life testing . Tool - life testing methods in turning tools

9- MATLAB 6.5