Universitatea Tehnică din Cluj-Napoca

Facultatea de Automatică și Calculatoare

Disciplina:

IDENTIFICAREA SISTEMELOR

Proiect:

IDENTIFICAREA UNEI AXE ACȚIONATE CU MOTOR BLDC

Profesor îndrumător:

Prof. Dr. Ing. Petru Dobra

Data 7 Ianuarie 2020

Student: FILIP ANDREI

Grupa: 30131

2019-2020

Cuprins

**1 Identificarea unei axe acționate cu motor BLDC – Noțiuni introductive proiect**

1.1 Obținerea datelor experimentale …………………………………...….3

1.1.1 Introducere ……………………………………………………..….…3

1.2 Achiziția datelor intrare-iesire ………………………………………...5

1.2,1 Desfășurarea experimentului ………………………………………...5

1.3 Procesarea datelor experimentale ……………………………………...5

1.3.1 Validarea modelului ……………………………………………….…5

1.3.2 Validarea rezultatelor …………………………………………….…..6

1.4 Metode si algoritmi pentru identificare …………………………….…..7

**2 Pregătirea datelor pentru identificare**

2.1 Achiziția datelor de intrare-ieșire.........................................................10

2.2 Interpolarea vitezei unghiulare ............................................................11

2.3 Decimarea datelor ................................................................................12

2.4 Alegerea datelor de identificare si verificare .......................................14

2.5 Cod Matlab ............................................................................................16

**3 Identificarea unui model care respecta testul de autocorelație**

3.1 Prin Metoda ARMAX rafinat cu PEM ………………………………...18

3.2 Prin Metoda Spațiul-Stărilor cu N4SID rafinat cu PEM……………….20

3.3 Cod Matlab……………………………………………………………..22

**4 Identificarea unui model care respecta testul de intercorelație**

4.1 Prin Metoda OE rafinat cu PEM ……………………………….............24

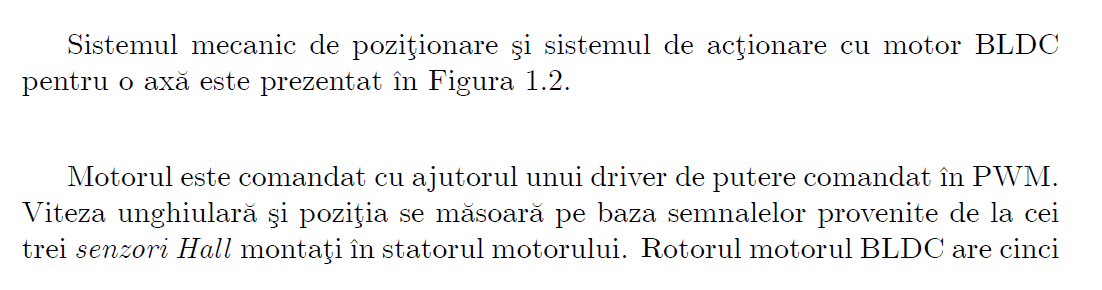
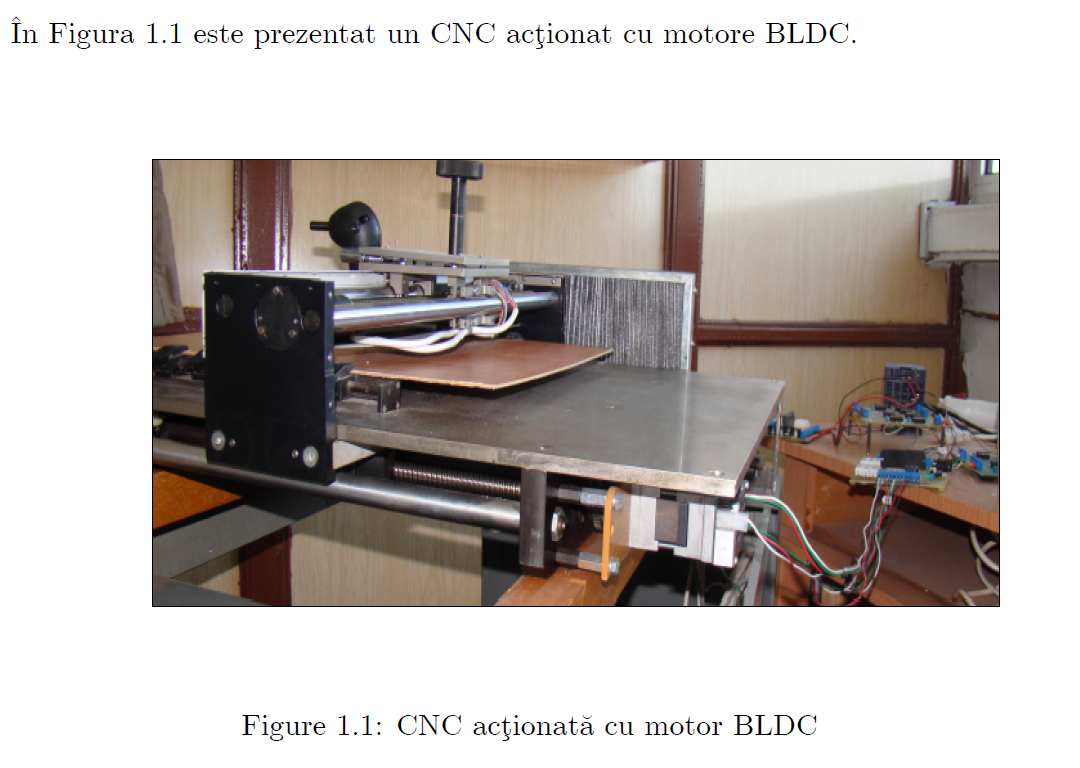
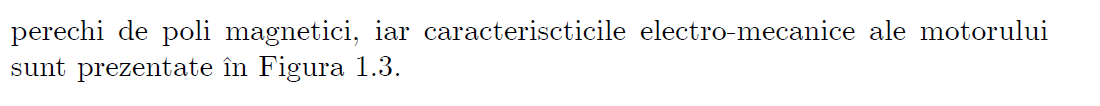
4.2 Prin Metoda Spațiul-Stărilor cu SSEST rafinat cu PEM……………….26

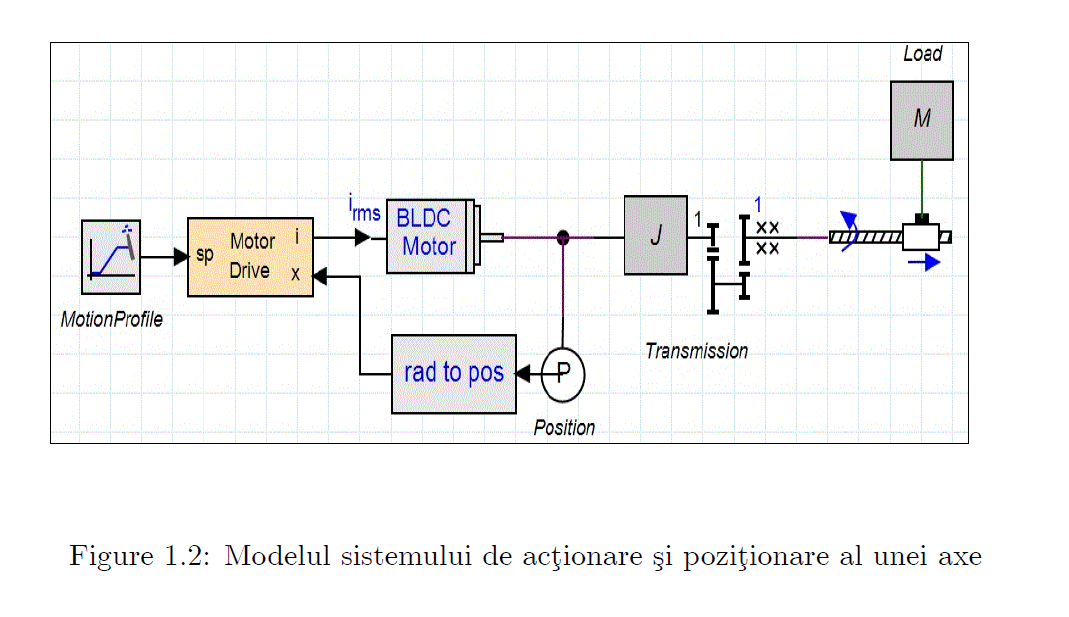
4.3 Combinarea celor 2 modele determinate..................................................28

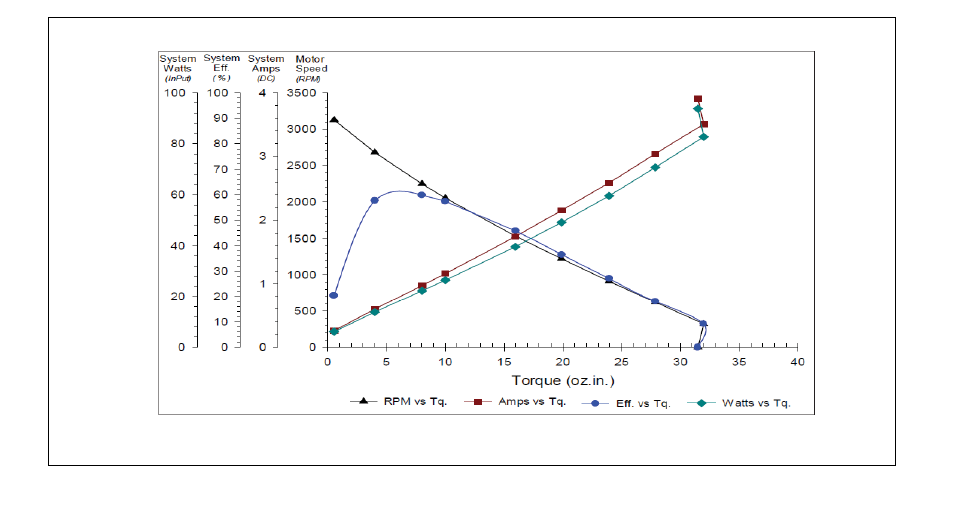
4.4 Cod Matlab………………………………………………………….…...29

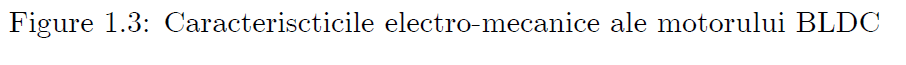
**Capitolul 1 Identificarea unui circuit electric**

* 1. **Obținerea datelor experimentale**
     1. **Introducere**



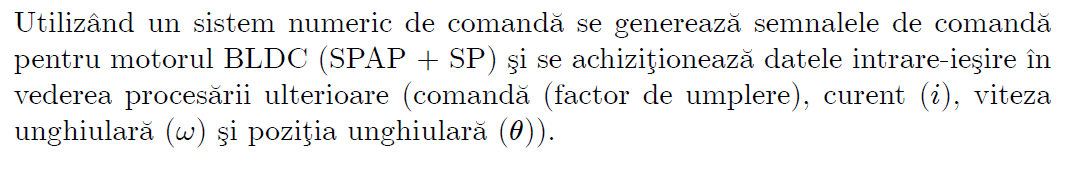




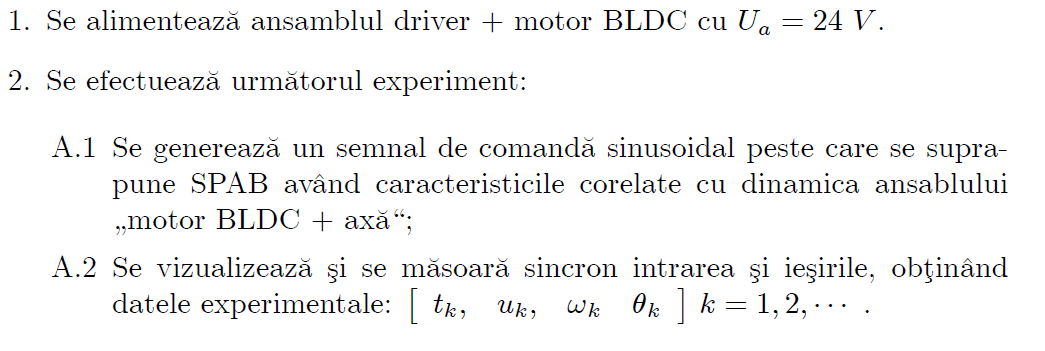


Aparatura utilizată: sursa de alimentare, multimetru, driver de putere, osciloscop, sistem numeric de comandă si achiziție a datelor.

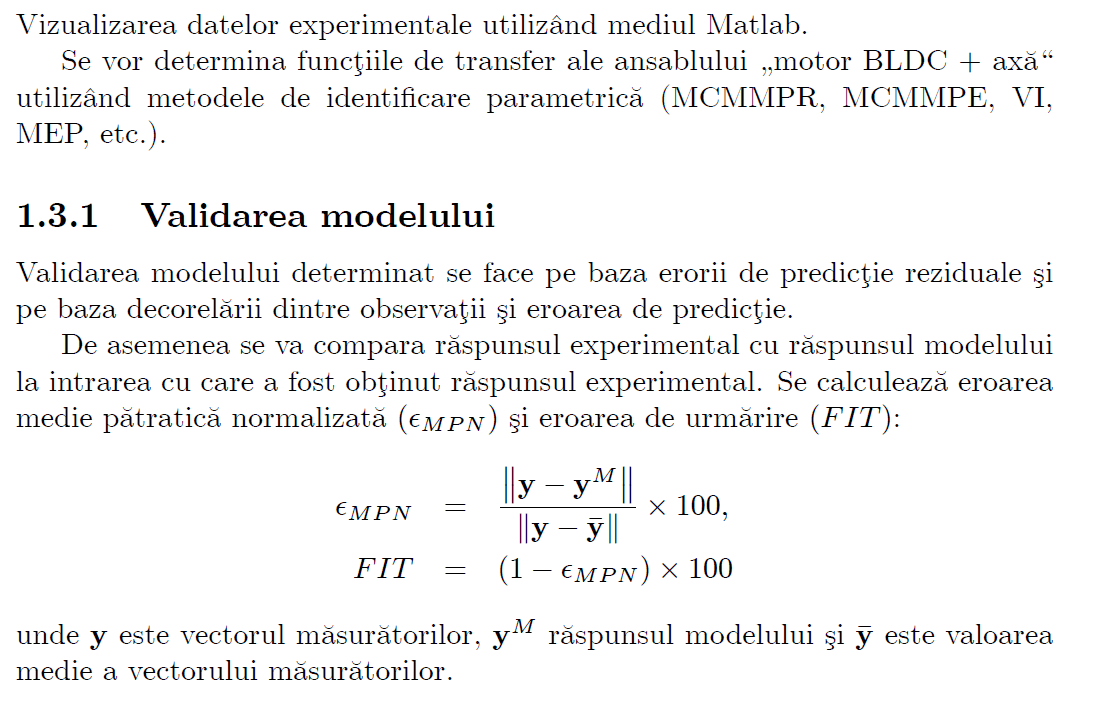
* 1. **Achiziția datelor de intrare-ieșire**



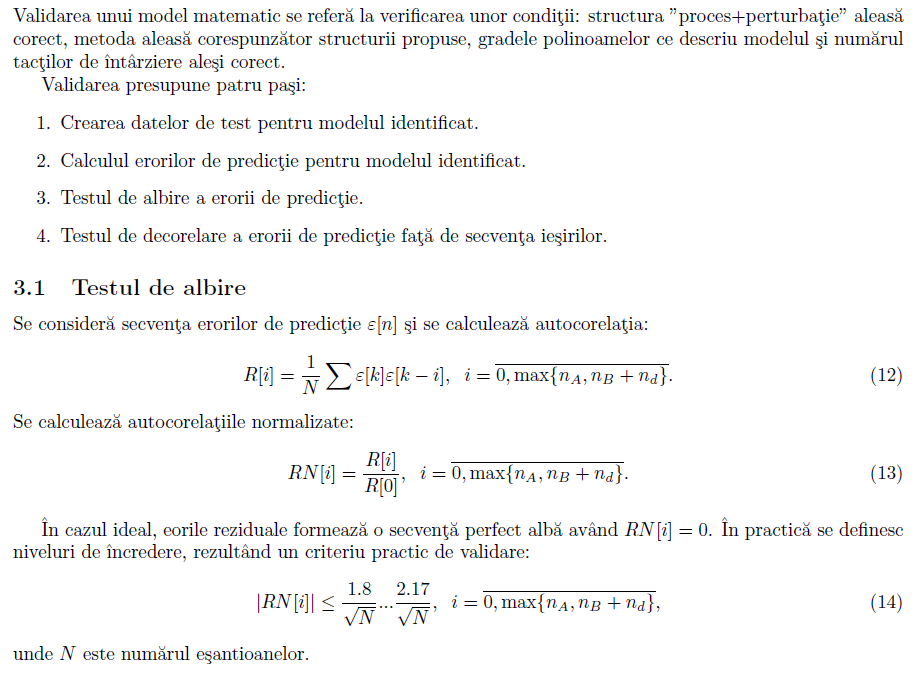
* + 1. **Achiziția datelor de intrare-ieșire**

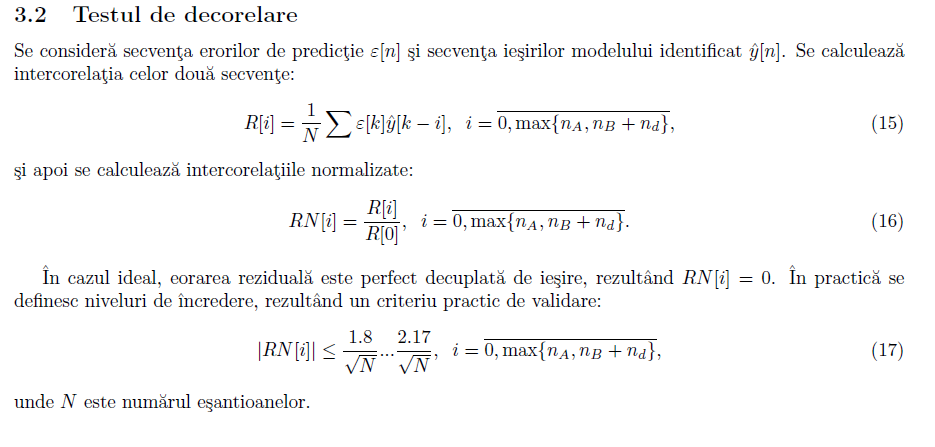


* 1. **Procesarea datelor experimentale**



* + 1. **Validarea rezultatelor**

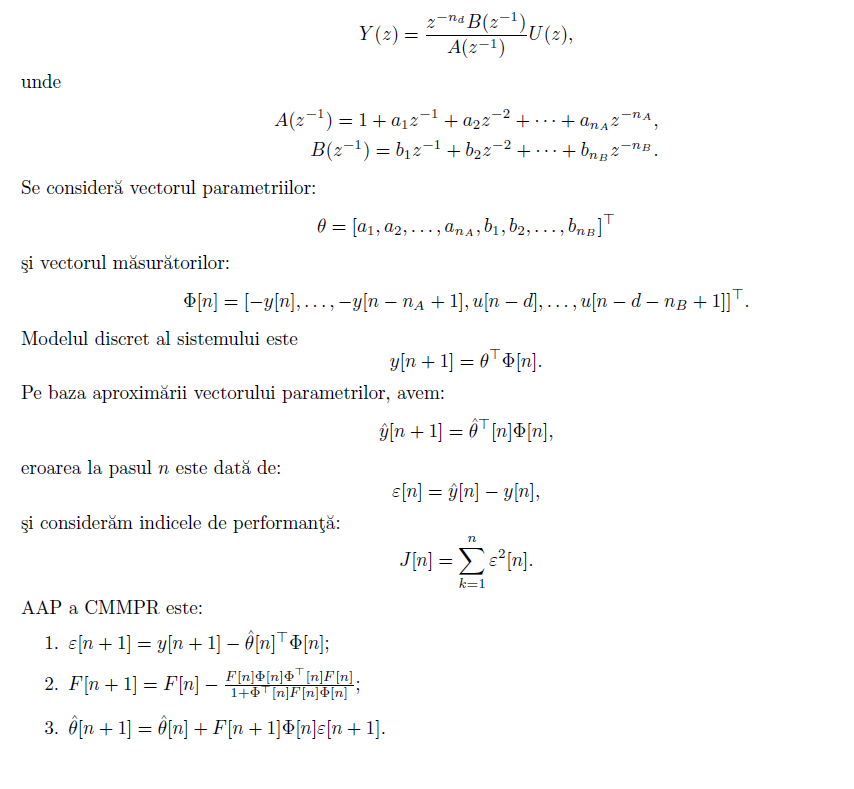




* 1. **Metode si algoritmi pentru identificare**

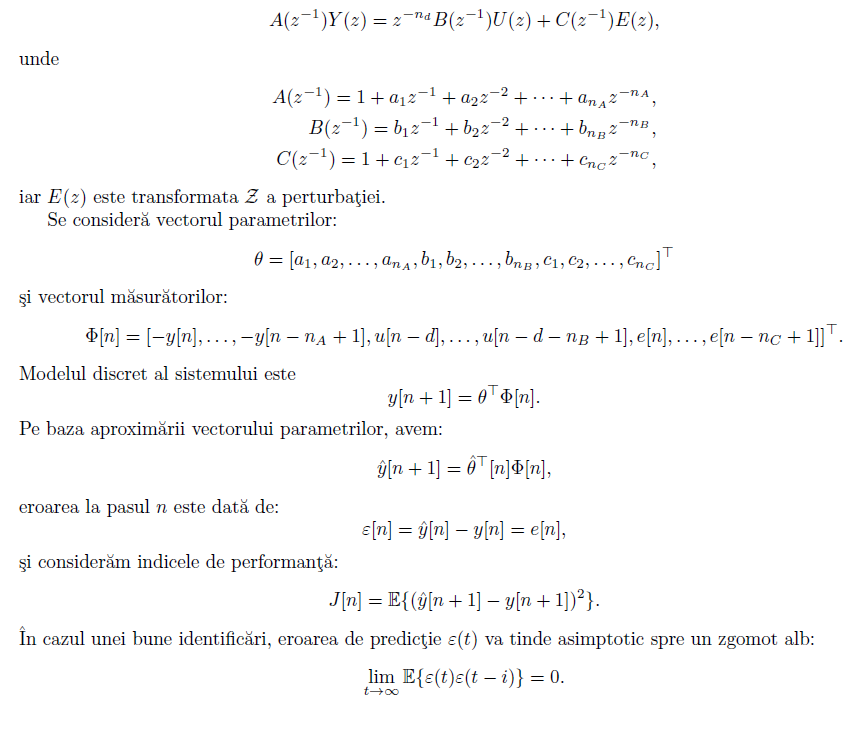
1. Algoritmul Celor Mai Mici Pătrate Recursive – ARX

Identificarea parametrică se face folosind funcția arx din toolbox-ul System Identification. Algoritmul de adaptare parametrica (AAP) a gradientului are ca obiectiv minimizarea unui criteriu patratic dependent de predicție. Se considera modelul discret unui sistem descris prin:



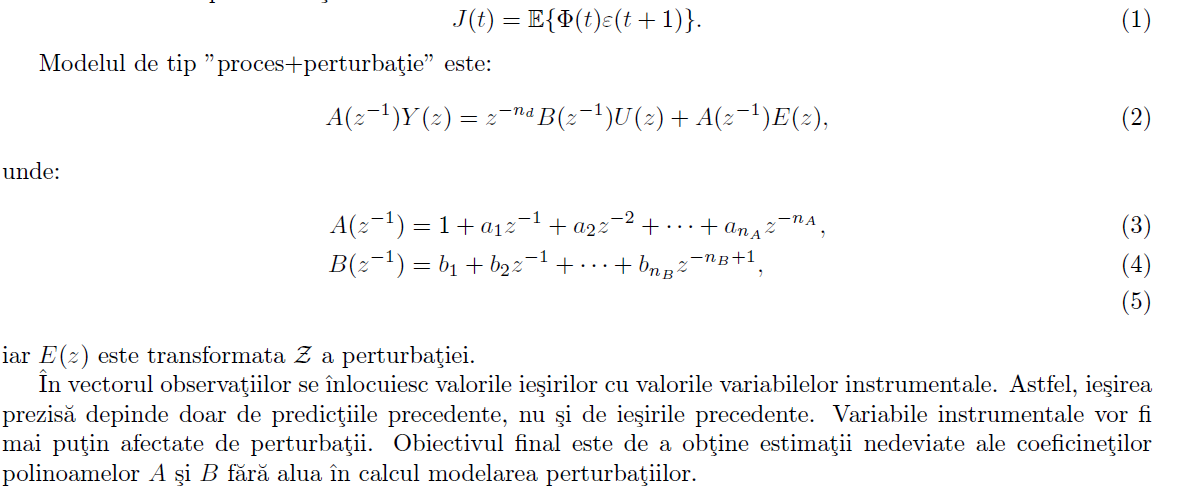
1. Metoda Celor Mai Mici Pătrate Extinsă – ARMAX

Identificarea parametrică se face folosind funcția armax din toolbox-ul System Identification. Metoda Celor Mai Mici Pătrate Extinsa (MCMMPE) a fost dezvoltată pentru identificarea unui sistem a cărui model discret este de tip “proces+perturbatie”:



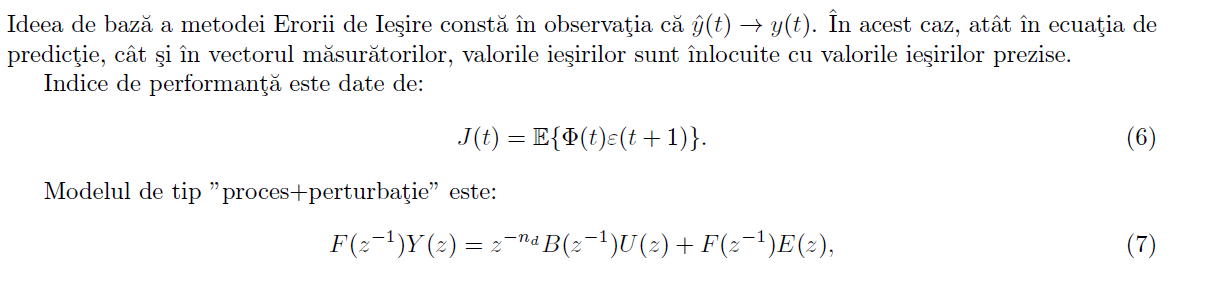
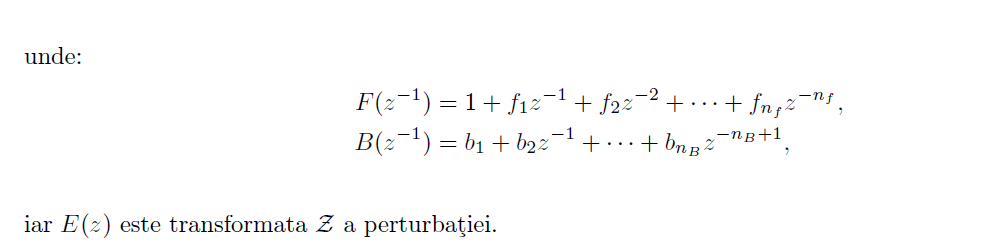
1. Metoda Variabilelor Instrumentale - IV

Identificarea parametrică se face folosind funcția iv4 din toolbox-ul System Identification. Metoda Variabilelor Instrumentale constă în crearea unui nou vector al observațiilor necorelat cu zgomotul, avand ca indice de performanta:



1. Metoda Erorii de Ieșire - OE

Identificarea parametrică se face folosind funcția oe din toolbox-ul System Identification.

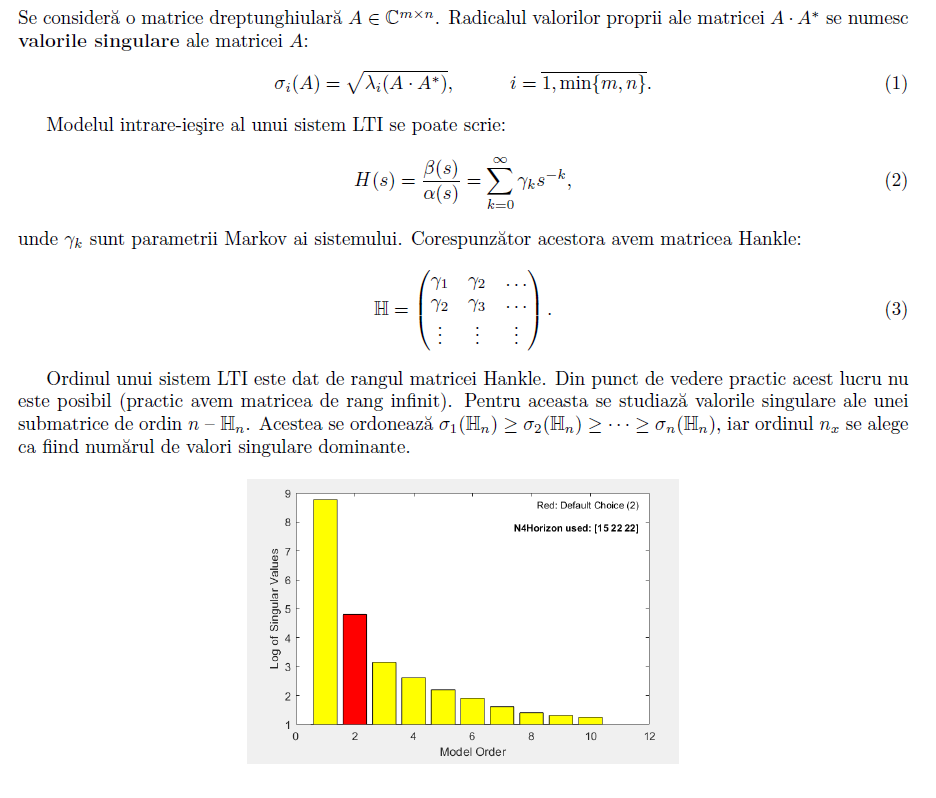


1. Metoda minimizării erorii de predicție - PEM

Identificarea parametrică se face folosind functia pem din toolbox-ul System Identification. Este un mecanism de rafinare a modelelor identificare folosind metodele parametrice descrise anterior. Se poate folosi si pentru identificarea unui model folosind doar funcția pem.

1. Estimarea modelelor de tip spațiul-stărilor

Identificarea se face folosind functia ssest sau n4sid din toolbox-ul System Identification.



**Capitolul 2**

**Pregătirea datelor pentru identificare**

**2.1 Achiziția datelor de intrare-ieșire**

Pentru achiziția datelor de intrare-ieșire importăm datele din fișierul *filip.mat,* cu ajutorul soft-ului *Matlab,* iar după plotarea acestora obținem graficul din figura 2.A .

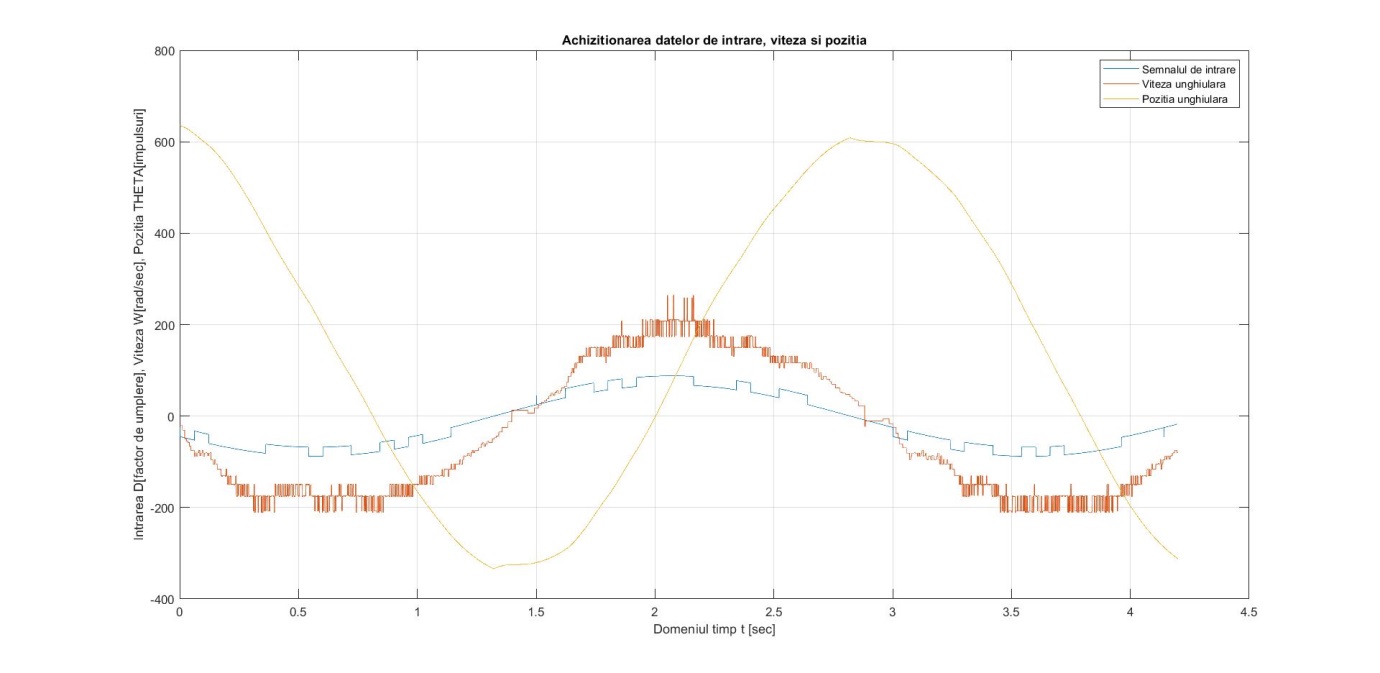


Figura 2.A. Achizitia datelor de intrare, de viteza si de pozitie

In această parte calculăm și perioada de eșantionare care o să ne fie utilă mai departe:

***Te = t(2) – t(1) = 4\*10^(-4) secunde = 0.4 milisecunde***

**2.2 Interpolarea vitezei unghiulare**

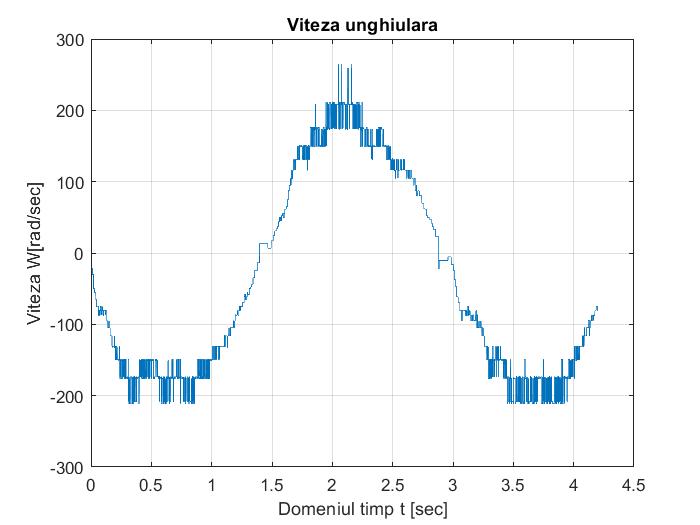
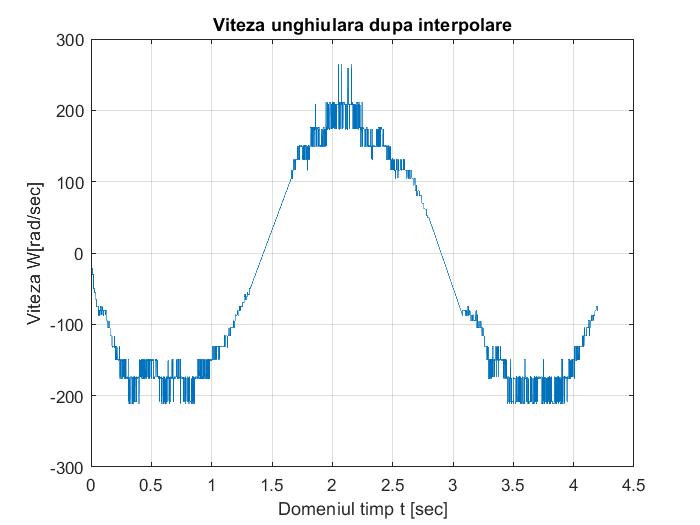
Din cauza lipsei de informatie de la semnalul de intrare, în cadrul semnalului de viteză unghiulară apar salturi bruște, așa cum se observă in figura 2.B. Pentru aceasta alegem porțiunile pe care vrem sa le interpolam, folosind niște indici bine aleși, urmând să obținem noua noastră viteză unghiulara interpolata pe care o vom folosi in continuare, vezi figura 2.C.

Figura 2.B

Viteza unghiulara



**i1=3300**

**i2=4150**

**i3=7000**

**i4=7700**

Figura 2.C

Viteza unghiulara

dupa interpolare

**2.3 Decimarea datelor**

Decimarea datelor ne va fi foarte utila mai departe astfel încât cu această metoda vom elimina datele care se repeta. Pentru aceasta ne uitam la viteza unghiulară interpolată si cautăm maximul acesteia. Din figura 2.D observam ca avem 3 maxime. Având în vedere că durata acestora este egala, alegem oricare vârf, si calculam „N”, care reprezinta pasul cu care ne vom construi noile date. Alegerea indicilor pentru calculul lui „N” este prezentat in fig. 2.E.

Figura 2.D. Cele trei varfuri ale vitezei unghiulare

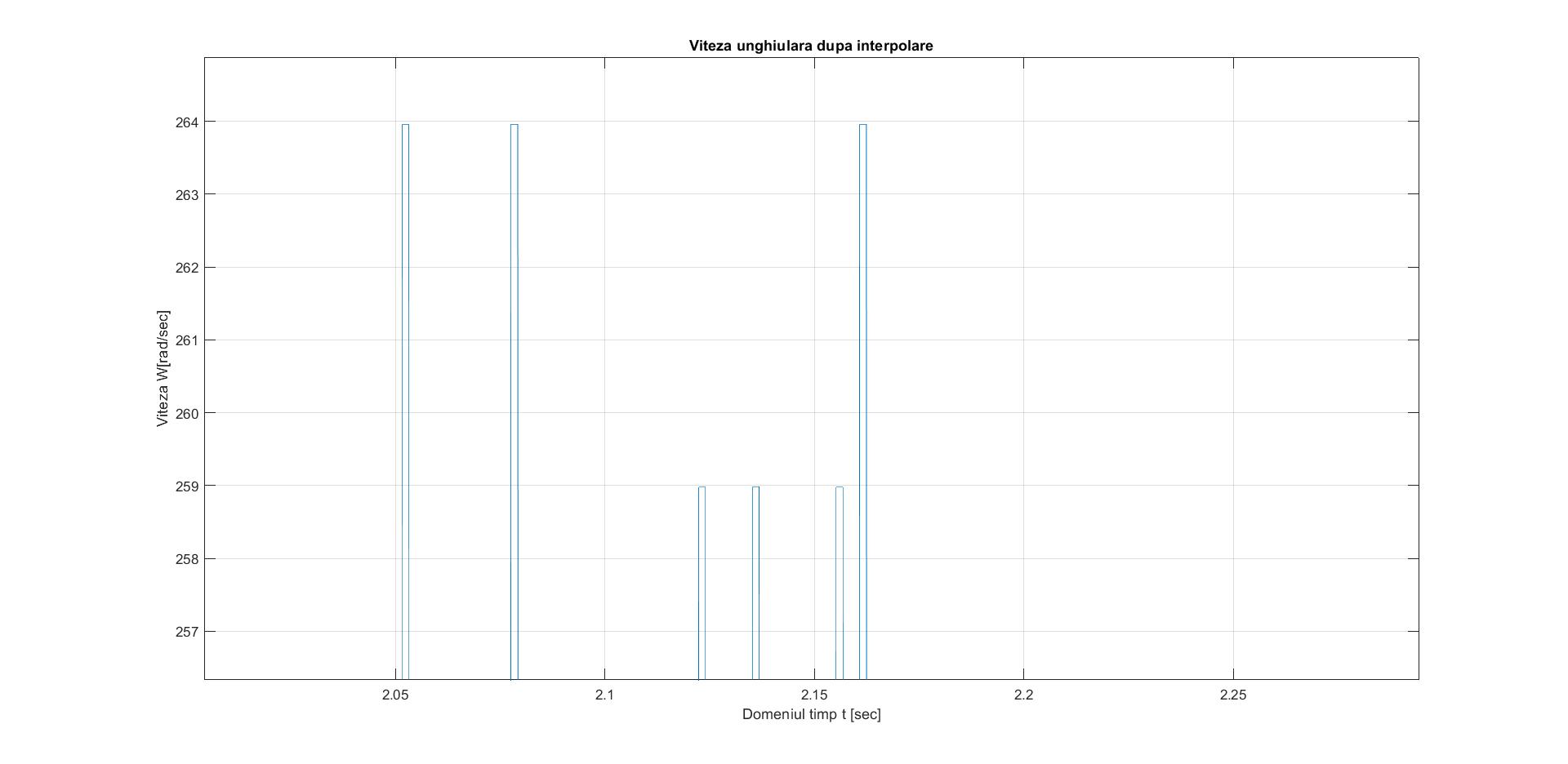
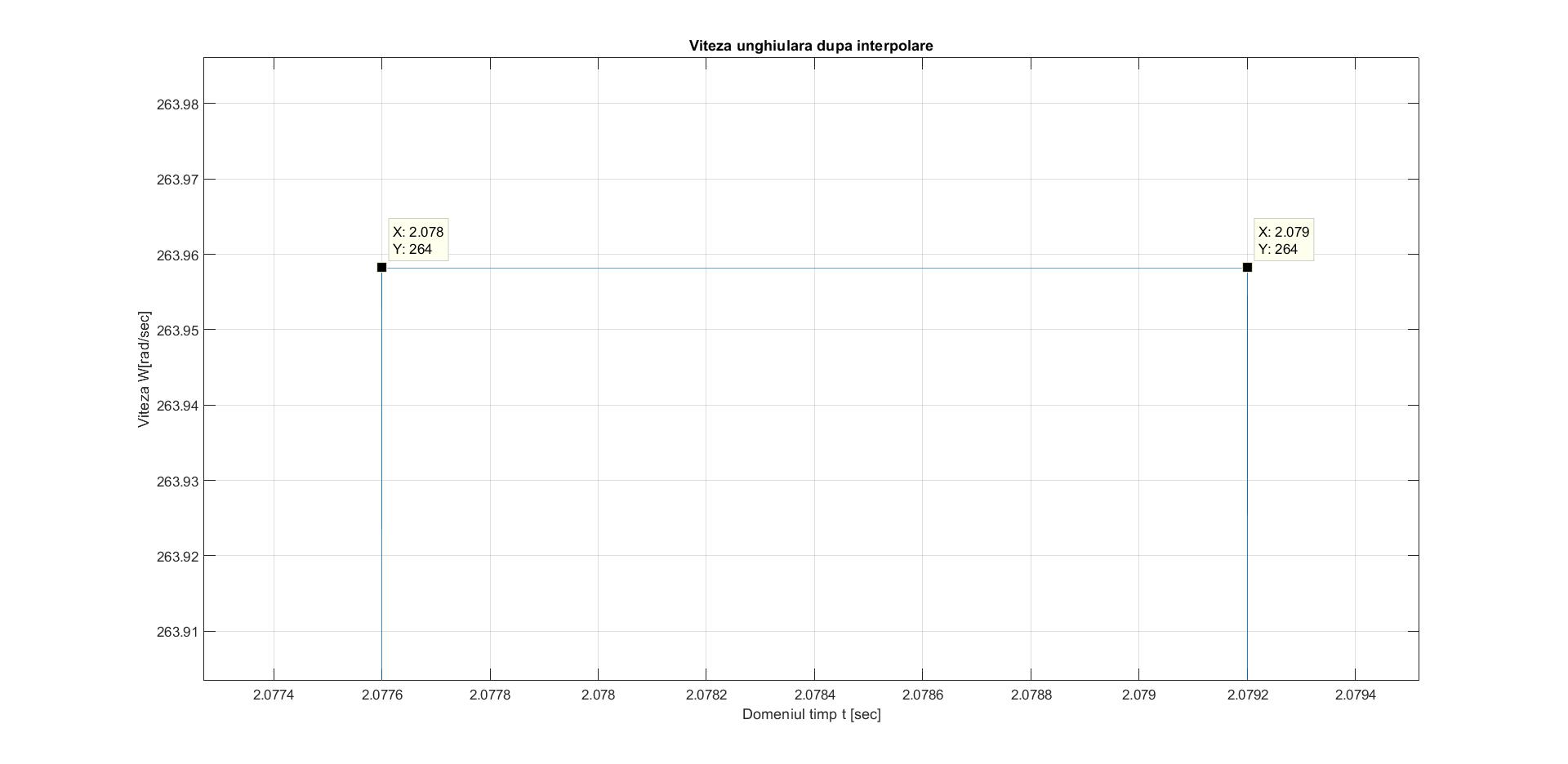


Figura 2.E. Alegerea indexilor pentru calculul lui „N”



**i7=5130**

**i8=5134**

După alegerea indicilor, noile date se construiesc astfel:

N=i8-i7+1; ***N=5;***

t\_dec=t(1:N:end);

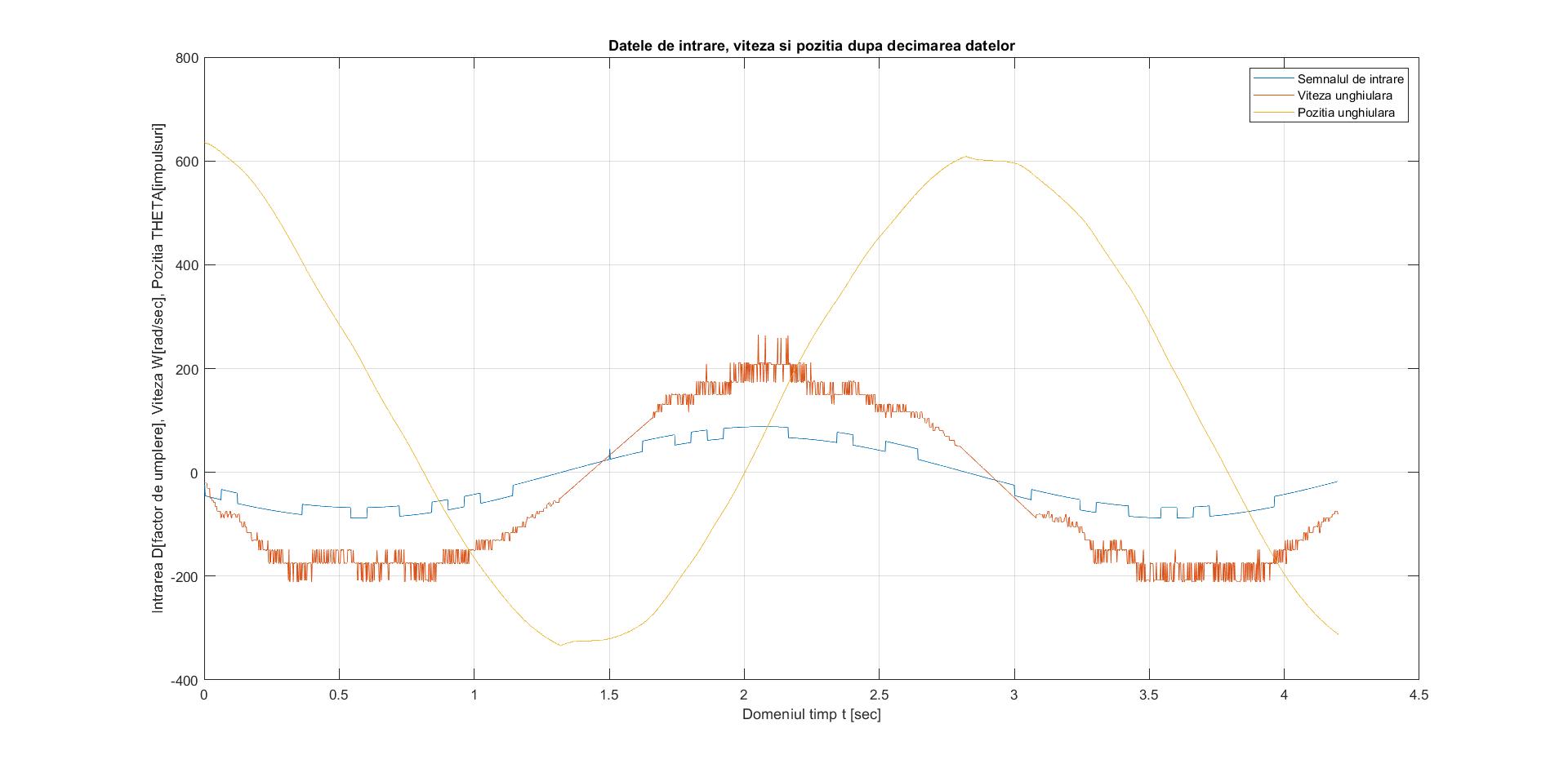
d\_dec=d(1:N:end);

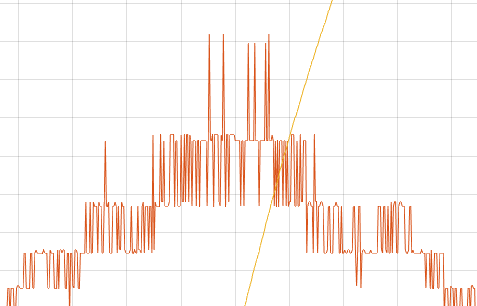
w\_dec=w(1:N:end);

theta\_dec=theta(1:N:end);

Te\_dec=Te\*N; ***Te\_dec=0.002 secunde***

Figura 2.F. Datele de intrare, viteză si poziția după decimare

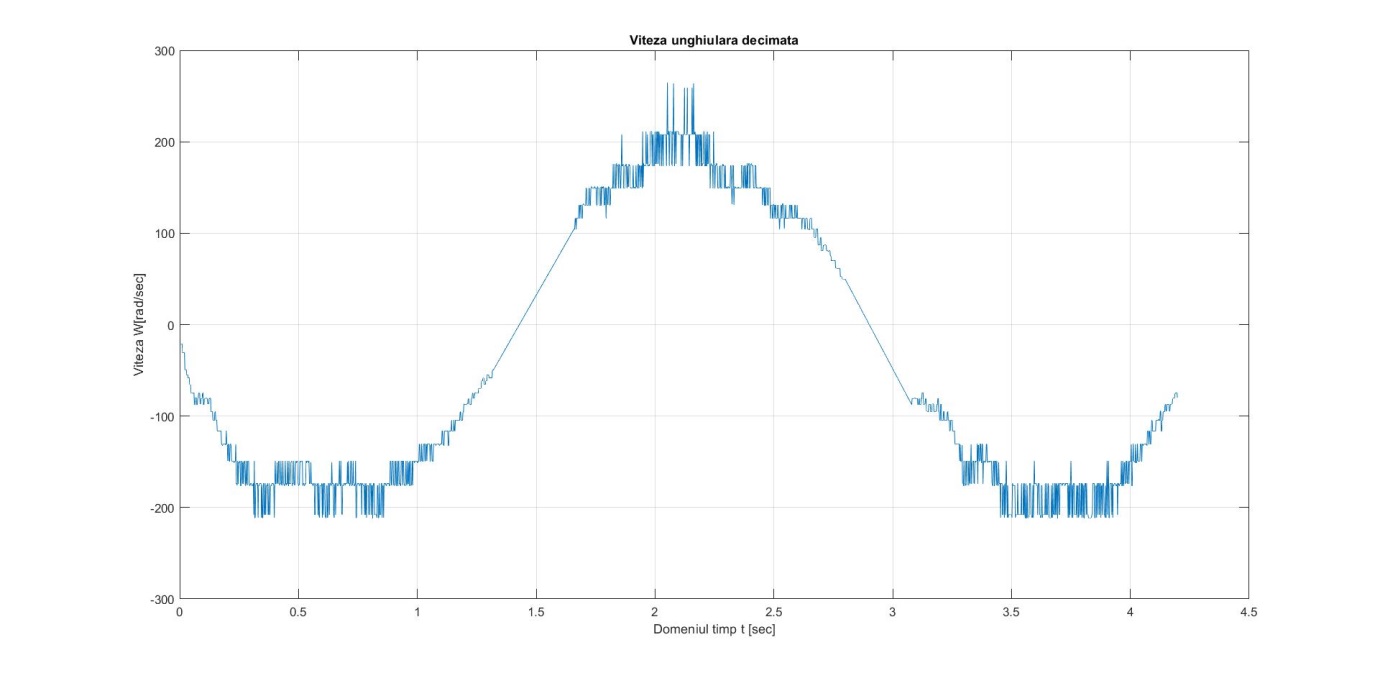




**2.4 Alegerea datelor de identificare și verificare**

Mai departe trebuie să ne separăm datele în date pentru identificarea modelului și date pentru verificarea acestora. Pentru aceasta avem nevoie de 4 indici bine aleși, pe care o sa îi alegem în funcție de sensul de rotație al motorului. Metoda de alegere a acestora este reprezentat în figura 2.G, pe reprezentarea vitezei unghiulare.

Figura 2.G. Alegerea indexilor de identificare si veificare



**t4=1162**

**t2=608**

**t1=66**

**t3=835**

În funcție de acesti indici ne declarăm structurile de identificare si verificare de la semnalul de intrare la viteza unghiulară, respectiv de la viteza unghiulară la poziția unghiulară. Acestea sunt necesare pentru validarea testelor de autocorelație si intercorelație. Mai este necesar să ne creăm structuri generale de date pentru testul de comparație.

%date identificare

data\_id\_w=iddata(w(t1:t2),d(t1:t2),Te);

data\_id\_th=iddata(theta(t1:t2),w(t1:t2),Te);

%date de verificare

data\_v\_w=iddata(w(t3:t4),d(t3:t4),Te);

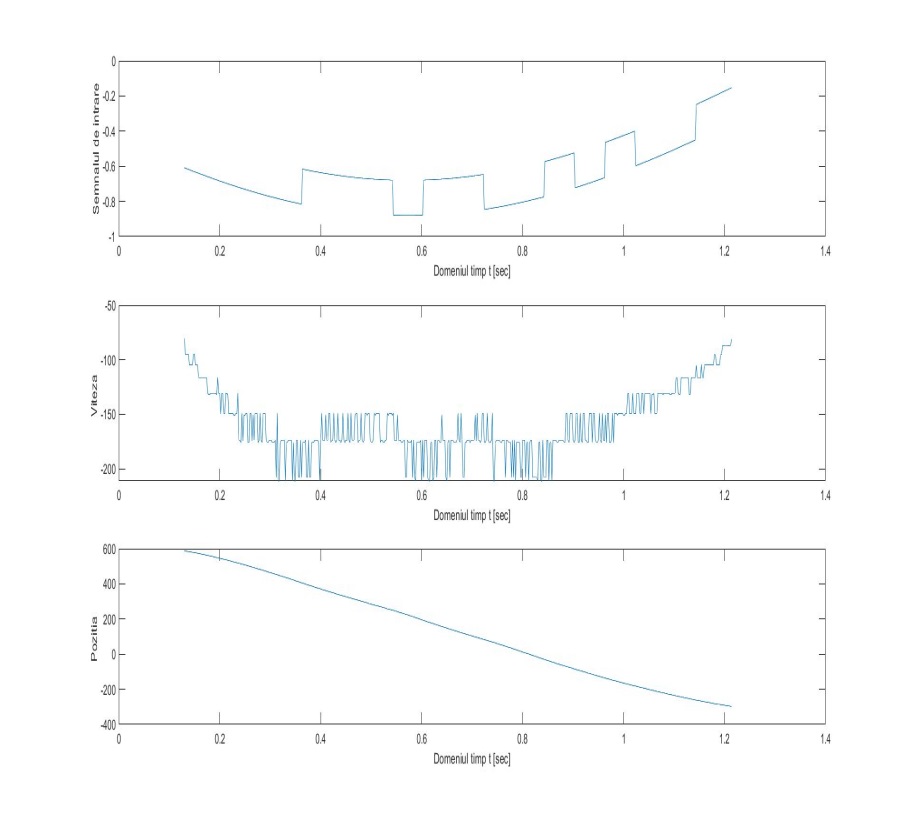
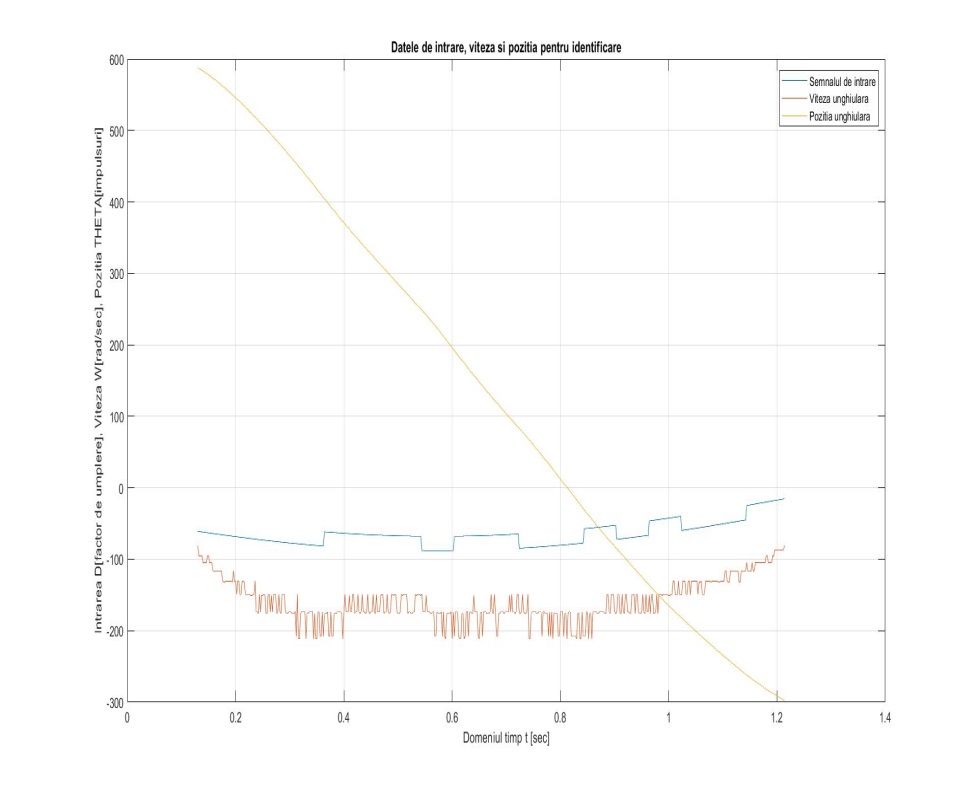
data\_v\_th=iddata(theta(t3:t4),w(t3:t4),Te);

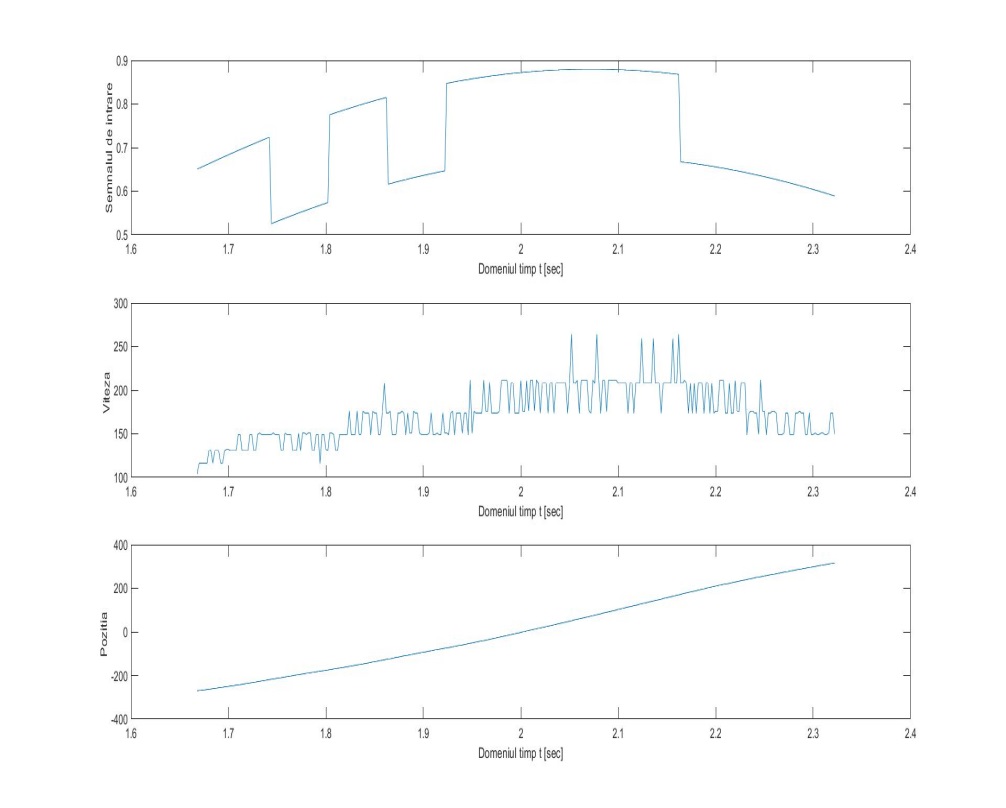
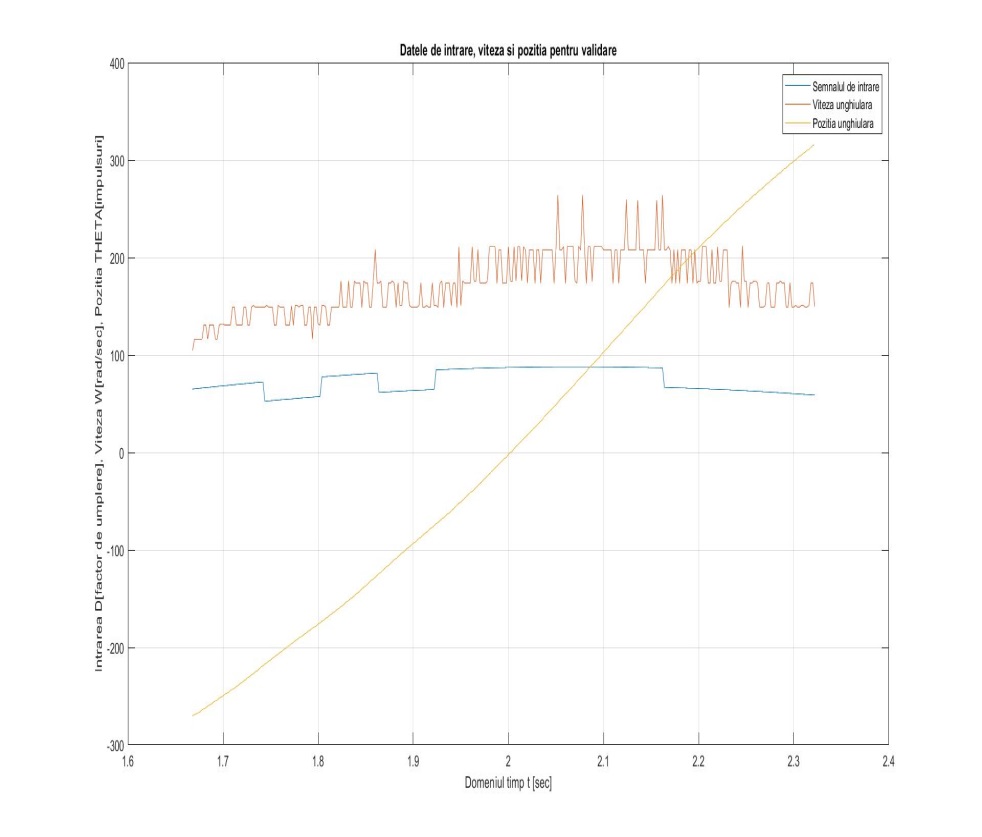
%date generale

data\_g\_w=iddata(w,d,Te);

data\_g\_th=iddata(theta,w,Te);

Figura 2.H. Datele pentru identificare



Figura 2.I. Datele pentru validare

**2.5 Cod Matlab**

%Achizitia datelor

t=double(filip.X.Data)';

theta=double(filip.Y(1,1).Data)';

w=double(filip.Y(1,2).Data)';

d=double(filip.Y(1,3).Data)';

%Plotarea datelor

plot(t, [d\*100 w theta]) ;grid;

legend('Semnalul de intrare','Viteza unghiulara','Pozitia unghiulara');

title('Achizitionarea datelor de intrare, viteza si pozitia');

xlabel('Domeniul timp t [sec]');

ylabel('Intrarea D[factor de umplere], Viteza W[rad/sec], Pozitia THETA[impulsuri]')

%Calculul perioadei de esantionare

Te=t(2)-t(1);

%Alegerea indicilor pentru interpolare

figure;plot(t, w);grid;

title('Viteza unghiulara');

xlabel('Domeniul timp t [sec]');

ylabel('Viteza W[rad/sec]')

i1=3300; i2=4150; i3=7000; i4=7700;

%Interpolarea vitezei

wi=w;

wi(i1:i2)=interp1([t(i1) t(i2)],[w(i1) w(i2)],t(i1:i2));

wi(i3:i4)=interp1([t(i3) t(i4)],[w(i3) w(i4)],t(i3:i4));

w=wi;

figure;plot(t, w);grid;

title('Viteza unghiulara dupa interpolare');

xlabel('Domeniul timp t [sec]');

ylabel('Viteza W[rad/sec]')

%Alegerea indicilor pentru decimare

i7=5130; i8=5134;

%Decimarea datelor

N=i8-i7+1

t\_dec=t(1:N:end);

d\_dec=d(1:N:end);

w\_dec=w(1:N:end);

theta\_dec=theta(1:N:end);

Te\_dec=Te\*N

t=t\_dec; d=d\_dec; w=w\_dec; theta=theta\_dec;

%Plotarea datelor dupa decimare

figure; plot(t, [d\*100 w theta]);grid;

legend('Semnalul de intrare','Viteza unghiulara','Pozitia unghiulara');

title('Datele de intrare, viteza si pozitia dupa decimarea datelor');

xlabel('Domeniul timp t [sec]');

ylabel('Intrarea D[factor de umplere], Viteza W[rad/sec], Pozitia THETA[impulsuri]')

figure;plot(t, w);grid;

title('Viteza unghiulara decimata');

xlabel('Domeniul timp t [sec]');

ylabel('Viteza W[rad/sec]')

%Alegerea indicilor pentru identificare si verificare

t1=66; t2=608; t3=835; t4=1162;

%date identificare

data\_id\_w=iddata(w(t1:t2),d(t1:t2),Te);

data\_id\_th=iddata(theta(t1:t2),w(t1:t2),Te);

%date de verificare

data\_v\_w=iddata(w(t3:t4),d(t3:t4),Te);

data\_v\_th=iddata(theta(t3:t4),w(t3:t4),Te);

%date generale

data\_g\_w=iddata(w,d,Te);

data\_g\_th=iddata(theta,w,Te);

%Plotarea datelor de identificare

figure; plot(t(t1:t2), [d(t1:t2)\*100 w(t1:t2) theta(t1:t2)]) ;

grid; legend('Semnalul de intrare','Viteza unghiulara','Pozitia unghiulara');

title('Datele de intrare, viteza si pozitia pentru identificare');

xlabel('Domeniul timp t [sec]');

ylabel('Intrarea D[factor de umplere], Viteza W[rad/sec], Pozitia THETA[impulsuri]')

figure; subplot(3,1,1); plot(t(t1:t2), d(t1:t2) ) ; xlabel('Domeniul timp t [sec]'); ylabel('Semnalul de intrare');

subplot(3,1,2); plot(t(t1:t2), w(t1:t2) ) ; xlabel('Domeniul timp t [sec]'); ylabel('Viteza ');

subplot(3,1,3); plot(t(t1:t2),theta(t1:t2)); xlabel('Domeniul timp t [sec]'); ylabel('Pozitia ');

%Plotarea datelor de validare

figure; plot(t(t3:t4), [d(t3:t4)\*100 w(t3:t4) theta(t3:t4)]) ;

grid; legend('Semnalul de intrare','Viteza unghiulara','Pozitia unghiulara');

title('Datele de intrare, viteza si pozitia pentru validare');

xlabel('Domeniul timp t [sec]');

ylabel('Intrarea D[factor de umplere], Viteza W[rad/sec], Pozitia THETA[impulsuri]')

figure; subplot(3,1,1); plot(t(t3:t4), d(t3:t4) ) ; xlabel('Domeniul timp t [sec]'); ylabel('Semnalul de intrare');

subplot(3,1,2); plot(t(t3:t4), w(t3:t4) ) ; xlabel('Domeniul timp t [sec]'); ylabel('Viteza ');

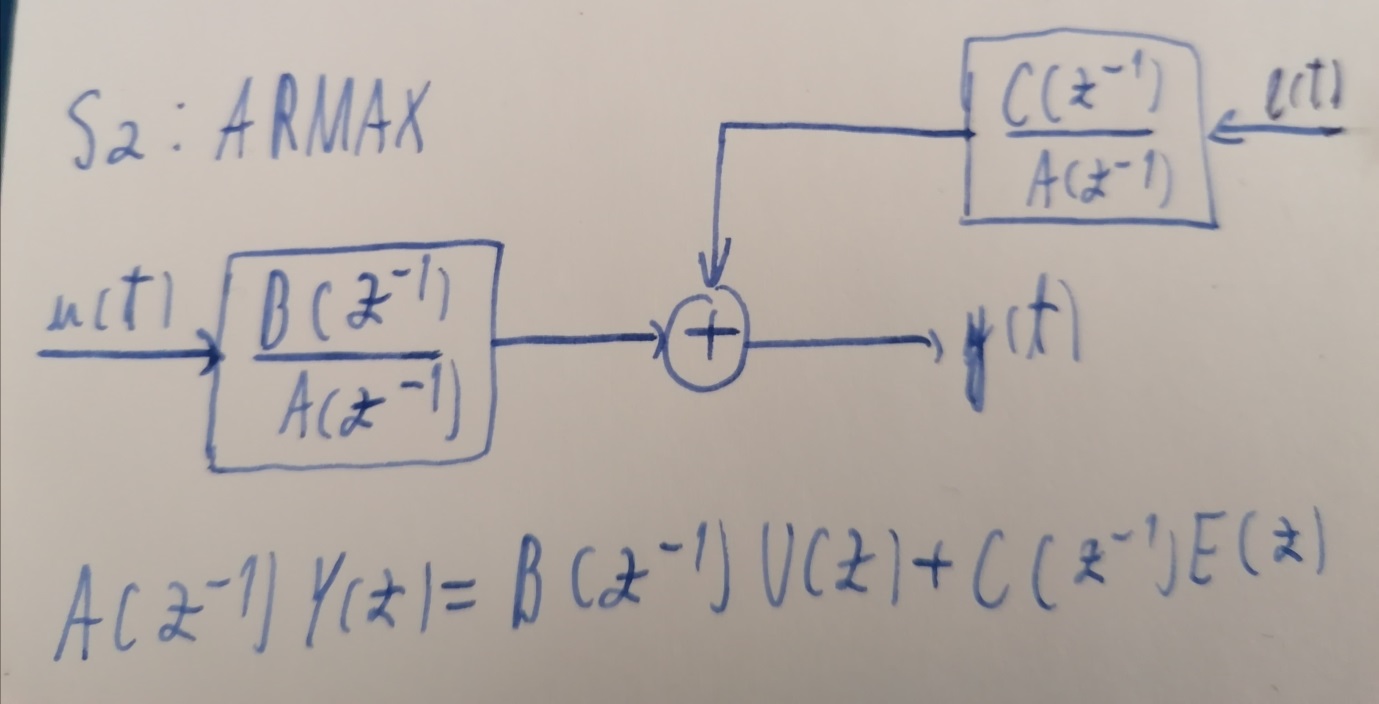
subplot(3,1,3); plot(t(t3:t4),theta(t3:t4)); xlabel('Domeniul timp t [sec]'); ylabel('Pozitia ');

**Capitolul 3**

**Identificarea unui model care respectă testul de autocorelație**

**3.1 Prin metoda ARMAX rafinat cu PEM**

Modelele obținute prin metoda ARMAX trebuie validate prin testul de autocorelație.



Structura pentru determinarea unui model în Matlab este următoarea:

Model = armax (DATA\_IDENTIFICARE , ORDERS)

ORDERS = [na nb nc nk] unde na,nb,nc,nk sunt parametrii de structură reprezentând gradele polinoamelor A,B,C respectiv „input delay” sau întârzierea.

Pentru modelul armax de la intrare la viteză am ales **na=2, nb=1, nc=1 si nk=1.**

Pentru modelul armax de la viteză la poziție am ales **na=1, nb=1, nc=1 si nk=1.**

Dupa obținerea acestora le-am rafinat prin metoda PEM, structura in Matlab fiind:

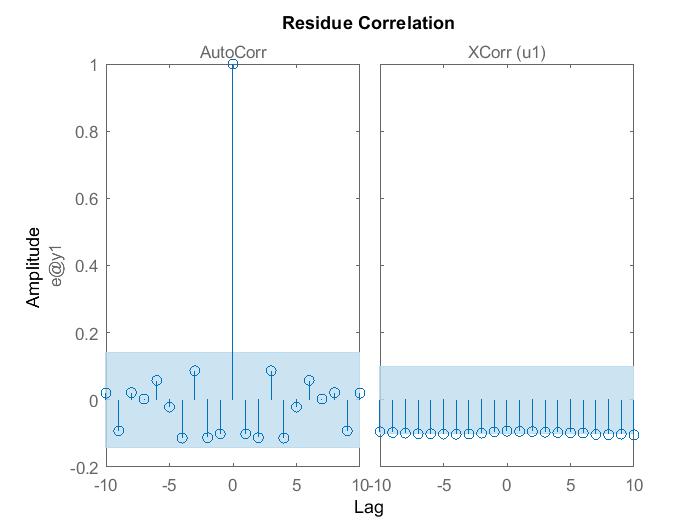
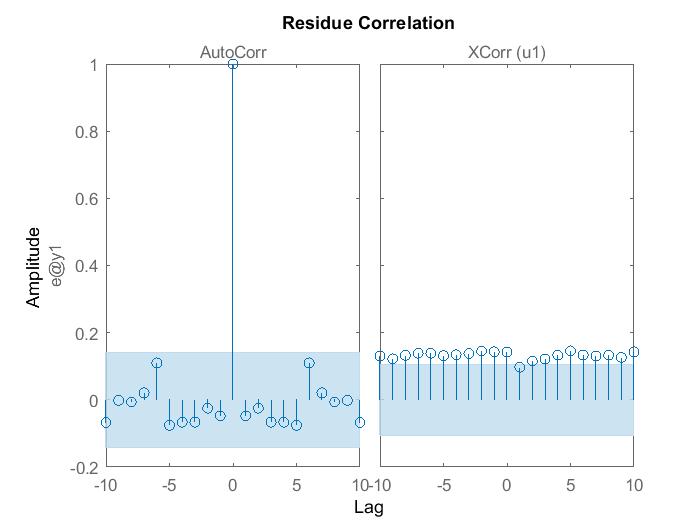
Model\_NOU = pem (DATA\_IDENTIFICARE , Model\_VECHI)

Pentru determinarea funcției de transfer apelăm următoarea structură:

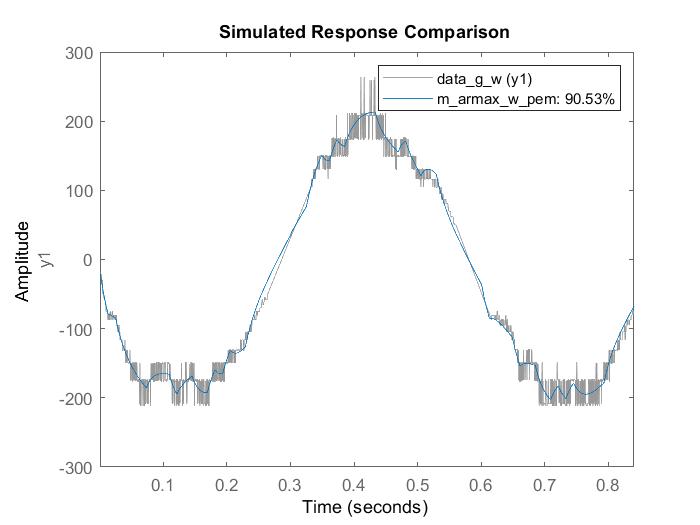
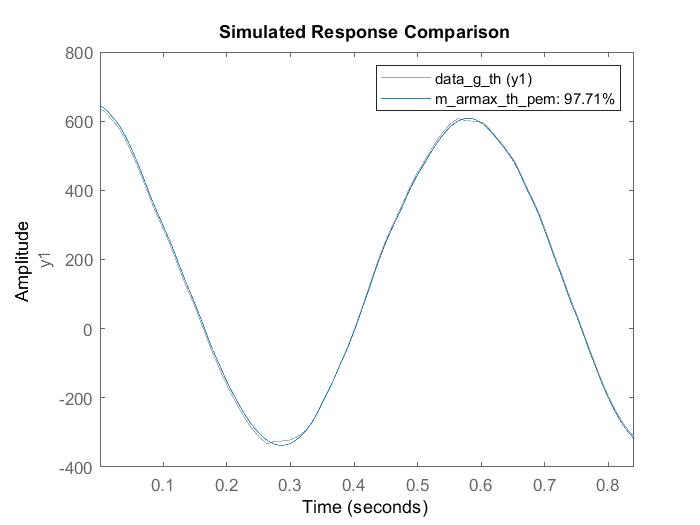
H=tf(model\_determinat), iar pentru transformarea in continuu apelam funcția d2c.

Mai jos putem vedea testele de autocorelație trecute (cele din partea stângă) pentru cele două modele de la intrare la viteză, respectiv de la viteză la poziție, în figurile 3.A1 si 3.A2, respectiv testele de comparație, în figurile 3.B1 si 3.B2

Figurile 3.A1 si 3.A2 reprezentând testele de corelație pentru cele 2 modele, cel din stânga pentru viteză, iar cel din dreapta pentru poziție

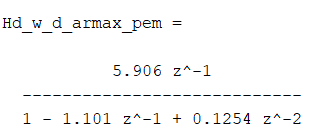
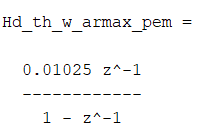
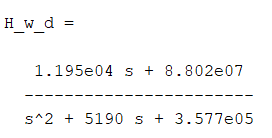
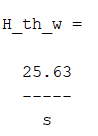


Figurile 3.B1 si 3.B2 reprezentând testele de comparație pentru cele 2 modele, cel din stânga pentru viteză, iar cel din dreapta pentru poziție



EMPN\_W=9.47% EMPN\_TH=2.29%

Funcțiile de transfer, atât în discret cât și în continuu, obținute sunt:

Pentru H\_w\_d avem un zero in -7.3657\*10^3 si doi poli in -5.1201\*10^3 si -0.0699\*10^3.

Modelul final obținut este:

**H(s) = [(3.063\*10^5\*s + 2.256\*10^9)/(s^2 + 5190\*s + 3.577\*10^5)]\*1/s**

**3.2 Prin metoda N4SID rafinat cu PEM**

In căutarea unui model mai accesibil, atât ordinul 1 pentru modelul de la intrare la viteză, cât și ordinul 1 de la viteză la poziție, care să respecte și testele de autocorelatțe, am încercat să determin un model cu metoda N4SID.

Prin comanda *n4sid(Data\_Identificare,1:10)*, ni se sugereaza gradul pentru fiecare model.



Structura pentru determinarea unui model în Matlab este următoarea:

Model = n4sid (DATA\_IDENTIFICARE , ORDER)

ORDER este parametrul de structură reprezentând gradul modelului de determinat

Pentru modelul n4sid de la intrare la viteza am ales gradul 1.

Pentru modelul n4sid de la viteza la pozitie am ales gradul 1.

După obținerea acestora le-am rafinat prin metoda PEM, structura în Matlab fiind:

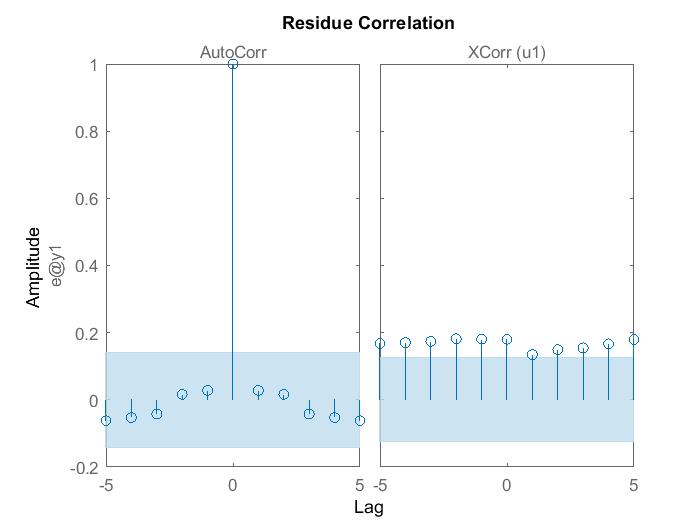
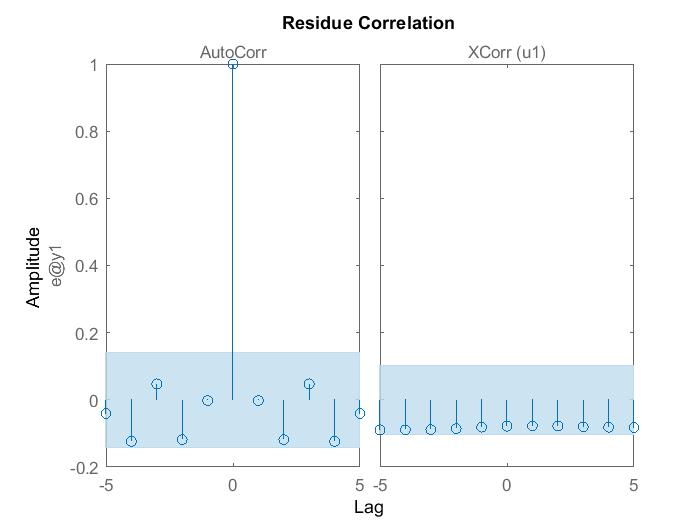
Model\_NOU = pem (DATA\_IDENTIFICARE , Model\_VECHI)

Pentru determinarea funcției de transfer apelăm următoarea structură:

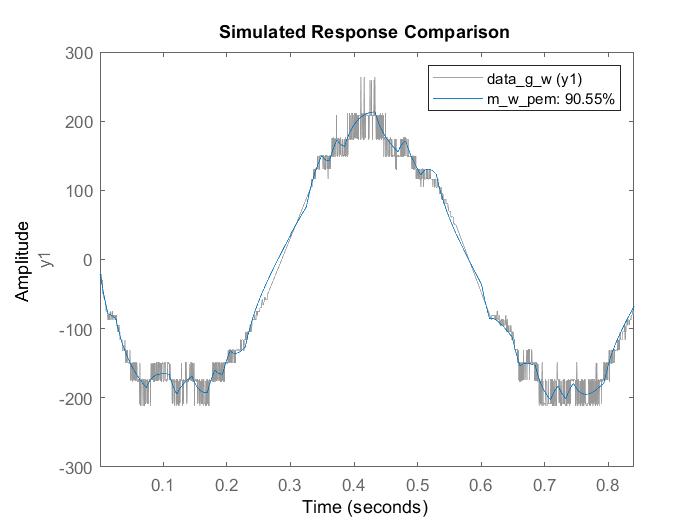
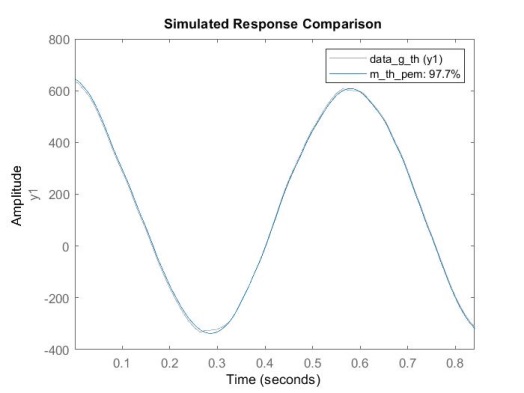
H=tf(model\_determinat), iar pentru transformarea în continuu apelăm funcția d2c.

Mai jos putem vedea testele de autocorelație trecute (cele din partea stângă) pentru cele doua modele de la intrare la viteză, respectiv de la viteză la poziție, în figurile 3.C1 si 3.C2, respectiv testele de comparație, în figurile 3.D1 si 3.D2

Figurile 3.C1 si 3.C2 reprezentând testele de corelație pentru cele 2 modele, cel din stânga pentru viteză, iar cel din dreapta pentru poziție

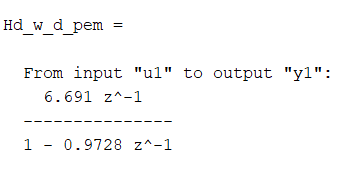
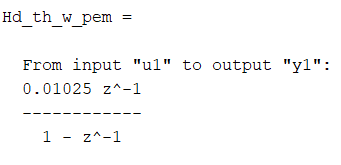
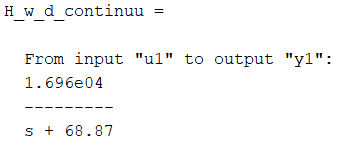
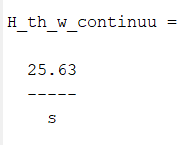


Figurile 3.D1 si 3.D2 reprezentând testele de comparație pentru cele 2 modele, cel din stânga pentru viteză, iar cel din dreapta pentru poziție



EMPN\_W=9.45% EMPN\_TH=2.3%

Funcțiile de transfer, atât în discret cât și în continuu, obținute sunt:

Pentru H\_w\_d\_continuu avem un pol in -68.87

Modelul final obținut este: **H(s) =[4.345\*10^5 / (s + 68.87)]\*1/s**

**Observatie: Acest model este mult mai simplu, dar cu performante asemanatoare fata de cel determinat cu Metoda ARMAX.**

**3.3 COD MATLAB**

%% ARMAX

m\_armax\_w=armax(data\_id\_w,[2 1 1 1]);

Hd\_w\_d=tf(m\_armax\_w.B,m\_armax\_w.A,Te,'variable','z^-1')

m\_armax\_th=armax(data\_id\_th,[1 1 1 1]);

Hd\_th\_w=tf(m\_armax\_th.B,m\_armax\_th.A,Te,'variable','z^-1')

figure

resid(data\_v\_w,m\_armax\_w,'corr',10)

figure

resid(data\_v\_th,m\_armax\_th,'corr',10)

figure

compare(m\_armax\_w,data\_g\_w)

figure

compare(m\_armax\_th,data\_g\_th)

H\_w\_d\_armax=d2c(Hd\_w\_d,'zoh')

H\_th\_w\_armax=d2c(Hd\_th\_w,'zoh')

%% ARMAX PEM

m\_armax\_w\_pem=pem(data\_id\_w,m\_armax\_w);

Hd\_w\_d\_armax\_pem=tf(m\_armax\_w\_pem.B,m\_armax\_w\_pem.A,Te,'variable','z^-1')

m\_armax\_th\_pem=pem(data\_id\_th,m\_armax\_th);

Hd\_th\_w\_armax\_pem=tf(m\_armax\_th\_pem.B,m\_armax\_th\_pem.A,Te,'variable','z^-1')

figure

resid(data\_v\_w,m\_armax\_w\_pem,'corr',10)

figure

resid(data\_v\_th,m\_armax\_th\_pem,'corr',10)

figure

compare(m\_armax\_w\_pem,data\_g\_w)

figure

compare(m\_armax\_th\_pem,data\_g\_th)

H\_w\_d=d2c(Hd\_w\_d\_armax\_pem,'zoh')

k=0.01025/Te;

H\_th\_w=tf([k],[1 0])

H\_final=series(H\_w\_d,H\_th\_w)

%% 1.Odinul sistemului

figure

n4sid(data\_id\_w,1:10)

figure

n4sid(data\_id\_th,1:10)

%% 2.N4SID

m\_w=n4sid(data\_id\_w,1);

Hd\_w\_d=tf(m\_w)

H1=d2c(Hd\_w\_d,'zoh')

m\_th=n4sid(data\_id\_th,1);

Hd\_th\_w=tf(m\_th)

H2=d2c(Hd\_th\_w,'zoh')

figure

resid(data\_v\_w,m\_w,'corr',5)

figure

resid(data\_v\_th,m\_th,'corr',5)

figure

compare(m\_w,data\_g\_w)

figure

compare(m\_th,data\_g\_th)

%% N4SID Pem

m\_w\_pem=pem(data\_id\_w,m\_w);

Hd\_w\_d\_pem=tf(m\_w\_pem)

figure

resid(m\_w\_pem,data\_v\_w,5);

figure

compare(m\_w\_pem,data\_g\_w);

m\_th\_pem=pem(data\_id\_th,m\_th);

Hd\_th\_w\_pem=tf(m\_th\_pem)

figure

resid(m\_th\_pem,data\_v\_th,5);

figure

compare(m\_th\_pem,data\_g\_th);

H\_w\_d\_continuu=d2c(Hd\_w\_d\_pem,'zoh')

k=0.01025/Te;

H\_th\_w\_continuu=tf([k],[1 0])

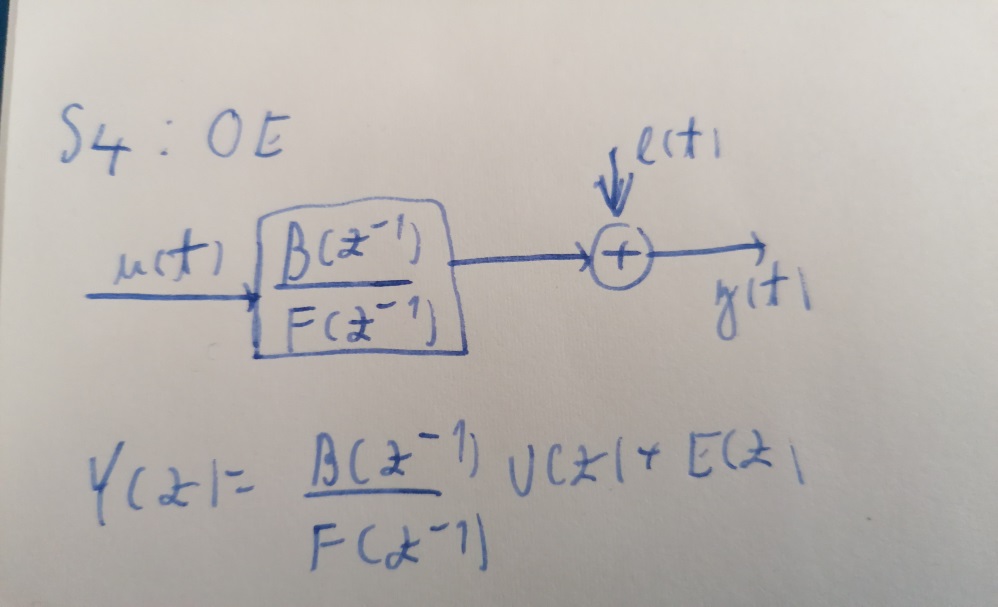
H\_final=series(H\_w\_d\_continuu,H\_th\_w\_continuu)

**Capitolul 4**

**Identificarea unui model care respectă testul de intercorelație**

**4.1 Prin metoda OE rafinat cu PEM**

Modelele obținute prin metoda OE trebuie validate prin testul de intercorelație.



Structura pentru determinarea unui model în Matlab este următoarea:

Model = oe (DATA\_IDENTIFICARE , ORDERS)

ORDERS = [ nb nf nk] unde nb,nf,nk sunt parametrii de structură reprezentând gradele polinoamelor B si F respectiv „input delay” sau întârzierea.

Pentru modelul armax de la intrare la viteză am ales **nb=1, nf=1 si nk=1.**

Pentru modelul armax de la viteză la poziție am ales **nb=1, nf=1 si nk=1.**

După obținerea acestora le-am rafinat prin metoda PEM, structura în Matlab fiind:

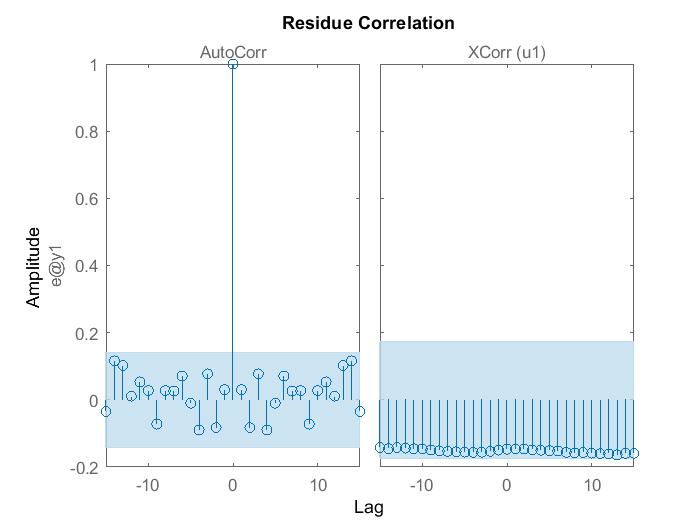
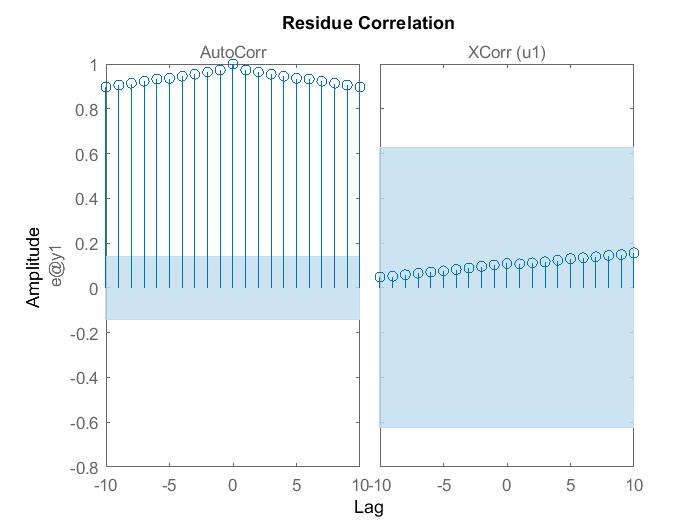
Model\_NOU = pem (DATA\_IDENTIFICARE , Model\_VECHI)

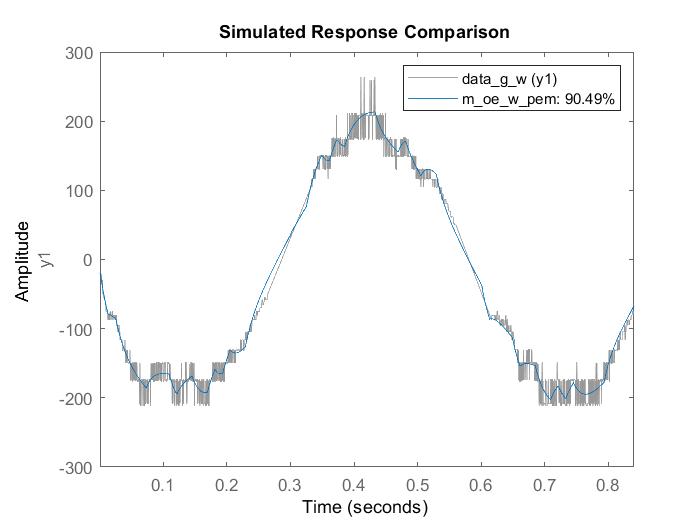
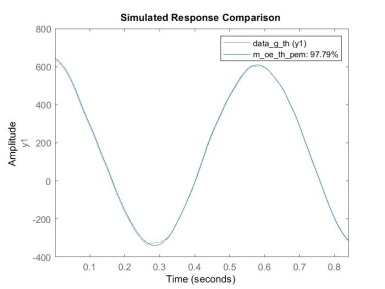
Pentru determinarea funcției de transfer apelăm următoarea structură:

H=tf(model\_determinat), iar pentru transformarea în continuu apelăm funcția d2c.

Mai jos putem vedea testele de intercorelație trecute (cele din partea dreaptă) pentru cele doua modele de la intrare la viteză, respectiv de la viteză la poziție, în figurile 4.A1 și 4.A2, respectiv testele de comparație, în figurile 4.B1 și 4.B2

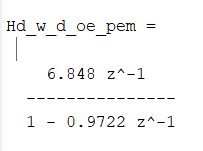
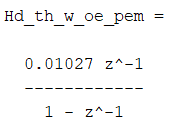
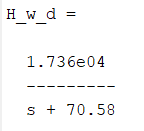
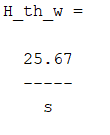
Figurile 4.A1 si 4.A2 reprezentând testele de corelație pentru cele 2 modele, cel din stânga pentru viteză, iar cel din dreapta pentru poziție



Figurile 4.B1 si 4.B2 reprezentând testele de comparație pentru cele 2 modele, cel din stânga pentru viteză, iar cel din dreapta pentru poziție

EMPN\_W=9.51% EMPN\_TH=2.21%

Funcțiile de transfer, atât în discret cât și în continuu, obținute sunt:

Pentru H\_w\_d avem un pol in -70.58.

Modelul final obținut este: **H(s) = [4.458\*10^5 / (s + 70.58)]\* 1/s**

**4.2 Prin metoda SSEST rafinat cu PEM**

În căutarea altui model accesibil, atât ordinul 1 pentru modelul de la intrare la viteză, cât și ordinul 1 de la viteză la poziție, care să respecte și testele de intercorelație, am încercat să determin un model cu metoda SSEST.

Prin comanda *ssest(Data\_Identificare,1:10)*, ni se sugereaza gradul pentru fiecare model.



Structura pentru determinarea unui model în Matlab este următoarea:

Model = ssest (DATA\_IDENTIFICARE , ORDER)

ORDER este parametrul de structură reprezentând gradul modelului de determinat

Pentru modelul ssest de la intrare la viteză am ales gradul 1.

Pentru modelul ssest de la viteză la poziție am ales gradul 1.

Dupa obținerea acestora le-am rafinat prin metoda PEM, structura în Matlab fiind:

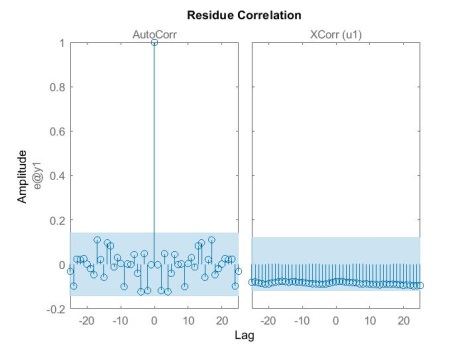
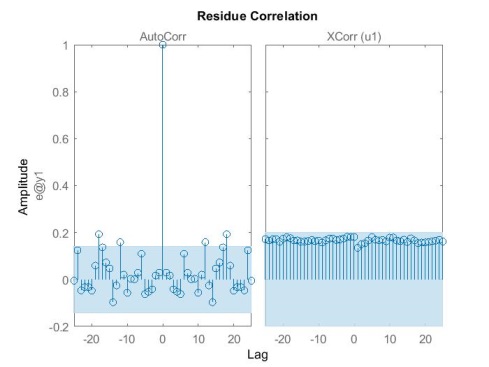
Model\_NOU = pem (DATA\_IDENTIFICARE , Model\_VECHI)

Pentru determinarea funcției de transfer apelăm următoarea structură:

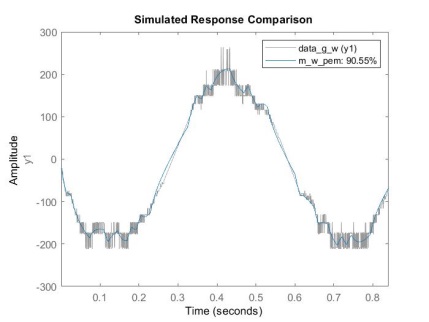
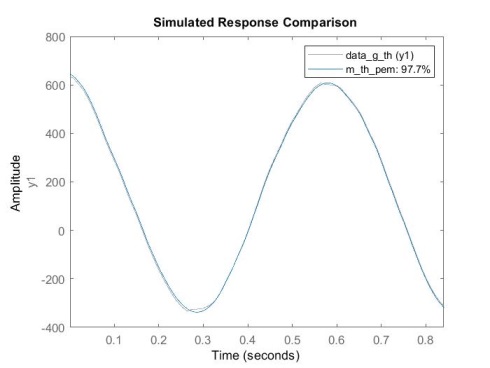
H=tf(model\_determinat), avantajul acestei metode este că le scoate direct in continuu.

Mai jos putem vedea testele de intercorelație trecute (cele din partea dreaptă) pentru cele două modele de la intrare la viteză, respectiv de la viteză la poziție, în figurile 4.C1 și 4.C2, respectiv testele de comparație, în figurile 4.D1 și 4.D2

Figurile 4.C1 si 4.C2 reprezentând testele de corelație pentru cele 2 modele, cel din stânga pentru viteză, iar cel din dreapta pentru poziție



Figurile 4.D1 si 4.D2 reprezentând testele de comparație pentru cele 2 modele, cel din stânga pentru viteză, iar cel din dreapta pentru poziție



EMPN\_W=9.45% EMPN\_TH=2.3%

Funcțiile de transfer, în continuu, obținute sunt:

H\_w\_d\_ssest\_pem(s) = 1.696\*10^4/(s+68.9) cu un pol în -68.9

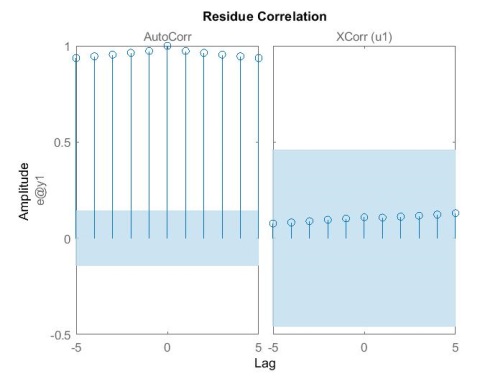
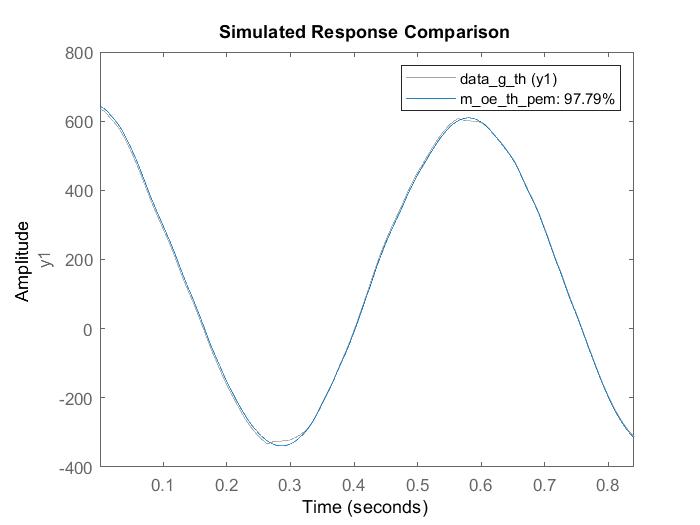
H\_th\_w\_ssest\_pem(s) = 25.62 / s

Modelul final obtinut este: **H(s) =[4.3452\*10^5 / (s + 68.9)]\*1/s**

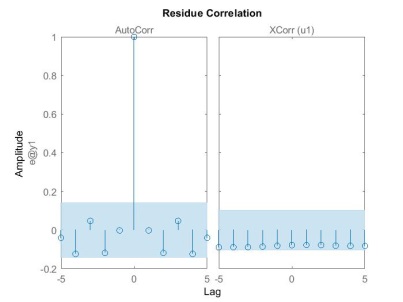
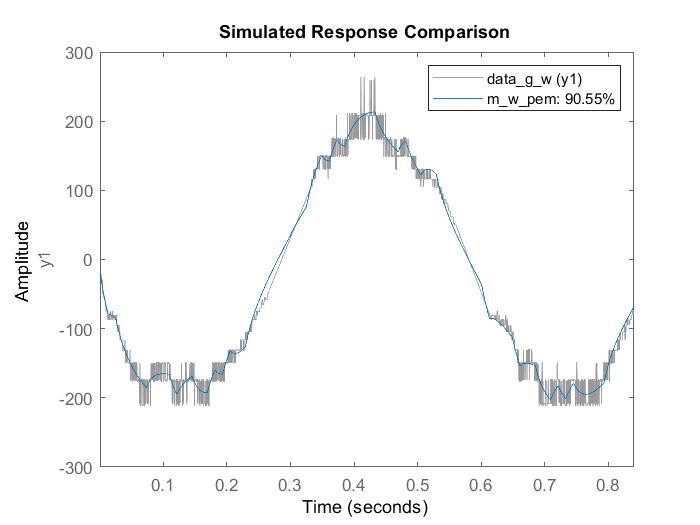
**4.3 Combinarea celor 2 modele determinate**

Așa cum se observă și mai sus, la modelul determinat cu Metoda OE rafinat cu PEM modelul de la intrare la viteză intră mai greu în banda de trecere, iar la modelul determinat cu SSEST rafinat cu PEM modelul de la viteză la poziție intră la limita în banda de trecere, la testul de intercorelație. Având în vedere că modelul de la viteză la poziție de la OE si modelul de la intrare la viteza de la SSEST par ideale, așa cum se observă și mai jos, o idee ar fi combinarea acestora obținând un model final mai practic care să respecte testul de intercorelație.

Figura 4.3.1 Modelul viteză-poziție OE+PEM



EMPN\_TH=2.21%

Figura 4.3.2 Modelul intrare-viteză SSEST+PEM

EMPN\_W=9.45%

Cu funcțiile de transfer aferente, determinate anterior:

H\_w\_d\_ssest\_pem(s) = 1.696\*10^4/(s+68.9) cu un pol in -68.9

H\_th\_w\_oe\_pem(s) = 25.67 / s

Modelul final obținut este: **H(s) =[4.353\*10^5 / (s + 68.9)]\*1/s**

**4.4 Cod MATLAB**

%% OE nb nf nk

m\_oe\_w=oe(data\_id\_w,[1 1 1]);

Hd\_w\_d\_oe=tf(m\_oe\_w.B,m\_oe\_w.F,Te,'variable','z^-1')

figure

resid(m\_oe\_w,data\_v\_w,'corr',15);

figure

compare(m\_oe\_w,data\_g\_w);

m\_oe\_th=oe(data\_id\_th,[1 1 1]);

Hd\_th\_w\_oe=tf(m\_oe\_th.B,m\_oe\_th.F,Te,'variable','z^-1')

figure

resid(m\_oe\_th,data\_v\_th,'corr',10);

figure

compare(m\_oe\_th,data\_g\_th);

%% OE Pem

m\_oe\_w\_pem=pem(data\_id\_w,m\_oe\_w);

Hd\_w\_d\_oe\_pem=tf(m\_oe\_w\_pem.B,m\_oe\_w\_pem.F,Te,'variable','z^-1')

figure

resid(m\_oe\_w\_pem,data\_v\_w,15);

figure

compare(m\_oe\_w\_pem,data\_g\_w);

m\_oe\_th\_pem=pem(data\_id\_th,m\_oe\_th);

Hd\_th\_w\_oe\_pem=tf(m\_oe\_th\_pem.B,m\_oe\_th\_pem.F,Te,'variable','z^-1')

figure

resid(m\_oe\_th\_pem,data\_v\_th,15);

figure

compare(m\_oe\_th\_pem,data\_g\_th);

H\_w\_d=d2c(Hd\_w\_d\_oe\_pem,'zoh')

Ki= 0.01027/Te;

H\_th\_w=tf(Ki,[1 0])

H\_final=series(H\_w\_d,H\_th\_w)

%% Odinul sistemului

figure

ssest(data\_id\_w,1:10)

figure

ssest(data\_id\_th,1:10)

%% SSEST

m\_w=ssest(data\_id\_w,1);

Hd\_w\_d=tf(m\_w)

m\_th=ssest(data\_id\_th,1);

Hd\_th\_w=tf(m\_th)

figure

resid(data\_v\_w,m\_w,'corr',5)

figure

resid(data\_v\_th,m\_th,'corr',5)

figure

compare(m\_w,data\_g\_w)

figure

compare(m\_th,data\_g\_th)

%% SSEST Pem

m\_w\_pem=pem(data\_id\_w,m\_w);

Hd\_w\_d\_pem=tf(m\_w\_pem)

figure

resid(m\_w\_pem,data\_v\_w);

figure

compare(m\_w\_pem,data\_g\_w);

m\_th\_pem=pem(data\_id\_th,m\_th);

Hd\_th\_w\_pem=tf(m\_th\_pem)

figure

resid(m\_th\_pem,data\_v\_th);

figure

compare(m\_th\_pem,data\_g\_th);