CUPRINS

LUCRAREA 1

IDENTIFICAREA SISTEMELOR FOLOSIND PACHETUL DE PROGRAME *MATLAB*. OBIECTE ŞI PROCEDURI UTILIZATE DE SUBPACHETUL *SYSTEM IDENTIFICATION TOOLBOX*

1. OBIECTIVELE LUCRĂRII

Familiarizarea cu facilităţile şi procedurile de identificare oferite de subpachetul *System Identification Toolbox* din pachetul de programe **MATLAB**. În etapa acoperită de lucrarea prezentă se explorează obiectele principale din subpachet asociate cu sistemele dinamice în reprezentări variate şi procedurile **ar, arx, armax**.

2. BREVIAR TEORETIC

Procedura de identificare experimentală a unui sistem se realizează în contextul unei lipse totale a informaţiei privitoare la forma modelului matematic, având la dispoziţie sistemul a cărei dinamică trebuie modelată. Sistemul este tratat ca *„black box”*, pentru care modelul se va sintetiza din informaţiile de tip I/E.

System Identification Toolbox – SIT – se referă la problema construirii modelelor matematice ale sistemelor dinamice bazate pe date experimentale. *SIT* este alcătuit dintr-o colecţie de tehnici de bază, bine înţelese şi verificate în aplicaţii practice.

Identificarea modelelor folosind datele experimentale presupune decizia factorului uman implicat în căutarea modelelor, ca şi prelucrarea acestor date în scopul furnizării bazei acestor decizii. De regulă, trebuie să fie parcurse mai multe bucle, revizuind deciziile anterioare. In acest context, softul interactiv reprezintă modalitatea logică de abordare practică a identificării sistemelor, fiind totodată şi un mijloc elegant de a „compacta" noţiunile teoretice extensive, făcându-le accesibile utilizatorului.

MATLAB este un mediu excelent pentru asemenea calcule interactive, date fiind conceptul său de workspace, facilităţile grafice şi modul de lucru cu datele. *SIT* este o colecţie de fişiere tip *.m* care implementează cele mai uzuale tehnici parametrice/neparametrice disponibile. Este, de asemenea, dedicat pentru utilizator, nu numai din punct de vedere al calculului, ci şi al evaluarii modelelor. Pentru automatişti, *SIT* este de foarte mare utilitate în problema identificării sistemelor.

SIT se afla în directorul **ident.off** al directorului principal **MATLAB**, fişierul demonstrativ putând fi accesat cu comanda **iddemo**.

O listă completă a funcţiilor pe care subpachetul de identificare le conţine se poate vizualiza, după o prealabilă invocare a platformei **MATLAB**, prin comanda

help toolbox/ident

care are ca efect afişarea listei respective cu numele funcţiilor pe categorii, însoţite de scurte explicaţii în limba engleză.

Detaliile asupra funcţiilor si funcţionării acestor programe se obţin prin comanda *help* urmată de numele programului/funcţiei**,** *type*, urmată de numele programului respectiv, sau *dbtype* urmată de numele programului şi de o pereche de întregi separaţii de **„:"** care indică linia de început şi linia de sfârşit a părţii din program afişate pe monitor.

2.1. Obiecte din subpachetul *System Identification Toolbox*

Prima secţiune din lucrare are ca scop familiarizarea cu funcţiile **MATLAB** care creează obiecte asociate cu sisteme dinamice diverse în variatele forme în care acestea se pot prezenta.

Se verifică pe rând şi apoi în relaţia lor logică funcţiile din subpachetul *System Identification Toolbox* şi din alte subpachete cu care programele de identificare au legătură.

Funcţia *tf* cu două argumente se studiază dupa cum urmează

```
num=[1 2];          % se iniţializează coeficienţii numărătorului funcţiei de
                    transfer.
den=[1 2 5];        % se iniţializează coeficienţii numitorului funcţiei de
                    transfer.
sis=tf(num,den)     % se generează un obiect de tipul tf care conţine
                    informaţia structurală şi parametrică a sistemului
                    dinamic modelat prin funcţia de transfer afişată.
```

Răspunsul la comanda ultimă (fără caracterul „;” la final) este:

transfer function:

```
s + 2
------------
s^2 + 2s +5
```

Se poate crea astfel un obiect a cărui structură se poate citi prin comanda *get(sis),*
cu rezultatul în mare parte uşor de interpretat:

```
num = {[0 1 2]}
den   = {[1 2 5]}
Variable = 's'
Ts = 0

Td = 0
InputName = {' '}
OutputName = {' '}
```

Notes = {}
UserData = []

Aceasta este convenabilă în lucrul cu sisteme dinamice, cu structuri şi parametrizări diverse. De exemplu, prin comanda *save nume fişier sis* se poate depune în memorie în fişierul *nume fişier.mat* informaţia din *sis* care poate fi oricând recuperată prin comanda *load nume fişier sis*.

În plus, prin apelurile simple la alte funcţii, cum sunt *impulse (sis)* sau *step(sis)* se pot obţine răspunsurile sistemului la intrările tipice - impulsul unitar şi treapta unitară.

Recuperarea „pe bucăţi" a informaţiei din obiectul cu numele *sis*, de tipul *tf* se obţine prin comenzile *[a,b] = tfdata (sis)* sau *[a,b] = tfdata(sis,'v')* cu rezultatul depunerii în a şi b a numărătorului şi numitorului funcţiei de transfer inclusă în *sis*, în varianta a doua sub forma vectorială explicită.

Modificarea unei părţi a unui obiect *tf* se poate face prin intermediul comenzilor de genul *set (sis,'numeproprietate',valoare).*

Un obiect din categoria *tf* poate servi ca argument multor altor funcţii cum sunt *bode, freqrest, nyquist, ltiview, nichols* etc. şi se recomandă încercarea lor şi urmărirea efectelor.

Aceeaşi funcţie *tf* poate crea obiecte asemănătoare, dar care se referă la sisteme discrete in timp. Pentru aceasta se include obligatoriu un al treilea argument care reprezintă intervalul de eşantionare:

sisd = tf (num,den,0.1);

Prin invocarea numelui rezultatului *sisd* se obţine conţinutul obiectului, adică forma funcţiei de transfer şi intervalul de discretizare. Variabila care apare în scrierea simbolică a funcţiei de transfer este *z*.

Dacă este vorba de modele continue în timp sau discrete în timp, există posibilitatea ca funcţia să fie scrisă în alte variabile cum ar fi *p* pentru reprezentări continue, z^{-1} sau *q* pentru reprezentări discrete.

Un alt gen de obiect de un tip diferit, tipul *ss*, se referă la modelele ecuaţie-de-stare-ecuaţie-de-observare. Secvenţa care urmează duce la crearea unui obiect de acest tip.

```
a = [-2 1;1 -3];
b = [1 1] ';
c = [1 0;0 1];
d = [0 0]';
sistem=ss (a,b,c,d);
```

Ultima comandă, fără caracterul suprimat al afişării ";" expune pe monitor cele patru matrici care intervin in forma modelului

$$\frac{dx}{dt} = Ax(t)+Bu(t) \qquad (1.1)$$
$$y(t) = Cx(t)+Du(t)$$

care ia în considerare evoluţia stării sistemului în forma continuă sau în forma discretă

$$x(t+\Delta t) = Ax(t)+Bu(t) \qquad (1.2)$$
$$y(t) = Cx(t)+Du(t)$$

dacă, din nou, se mai adaugă funcţiei *ss* încă un argument care este intervalul de eşantionare. Se intelege că în varianta din urmă timpul nu poate fi decât multiplu întreg al intervalului Δt.

Cu un obiect de acest tip, cum este obiectul sistem, se pot evalua răspunsurile sistemului la intrări variate, cu aceleaşi comenzi utilizate şi cu obiectul *sis*, de un alt tip, de tipul *tf*.

Matricile se pot recupera pe rând si separat prin comenzi de genul

```
al = sistem.a;
```

Foarte populară printre automatişti este descrierea unui sistem prin zerourile, polii si amplificarea/amplificările lui. Pentru aceasta, *System*

Identification Toolbox include o funcţie *zpk* destinata creării de obiecte de un tip diferit de cele descrise sumar mai devreme.

Sintaxa acestei funcţii este *sys=zpk(z,p,k)* sau *sys=zpk(z,p,k,t)* din nou după cum este vorba de reprezentări continue sau discrete in timp.

Dacă sistemul este de tipul *o intrare-o ieşire* (*SISO- Single Input-Single Output*), atunci zerourile şi polii se introduc ca vectori (vectorul vid uneori!), iar amplificarea ca un scalar.

Dacă sistemul este de tip *MIMO (Multiple Inputs-multiple Outputs), cu mai multe intrări şi mai multe ieşiri*, atunci argumentele *z* şi *p* sunt matrici celulare cu câte o celulă de vectori *z[i, j],p[i, j]* pentru fiecare intrare (*j*)-ieşire (*i*), iar *k* este o matrice rectangulară care conţine amplificările, de asemenea pentru fiecare canal intrare-ieşire.

Exemplu

Comanda

sist=zpk {{[];[2 3]}, {-1;[0 –1]},[-5;1]}

este pentru crearea obiectului /sistemului *sist* cu o intrare şi doua ieşiri

$$\begin{bmatrix} -5/(s+1) \\ (s-2)\ (s-3)/s(s+1) \end{bmatrix}$$

Deoarece în situaţii diferite pot fi necesare modele ale sistemelor în forme diferite, există un număr de funcţii care permit transformarea modelelor de un anumit tip în modele de un alt tip.

Astfel:

- Cu comenzile *ss2tf* şi *tf2ss* cu argumente potrivite se pot converti modelele de tip ecuaţie-de-stare – ecuaţie-de-observare în modele de tip funcţie de transfer şi invers.

- Cu comenzile *zp2tf* şi *tf2zp* cu argumente potrivite se pot converti modelele de tip poli-zerouri şi invers.
- Cu comenzile *ss2zp* si *zp2ss* cu argumente potrivite se pot converti modelele de tip ecuaţie-de-stare – ecuaţie-de-observare în modele de tip poli-zerouri şi invers.
- Comenzile *c2d* si *d2c* permit trecerea de la modele/sisteme continue în timp la modele/sisteme discrete în timp.

Desigur, aici nu se pot da toate detaliile referitoare la funcţiile din *System Identification Toolbox.*. Prin comenzile specifice *help* se pot observa şi studia mai în amănunt funcţiile invocate. Exemple simple tratate cu funcţiile subpachetului sunt, de asemenea, recomandate.

O altă formă în care se stochează şi se manevrează structuri şi parametri de modele este *forma/obiectul theta*.

O comandă *help theta* dezvăluie structura acestui obiect care este o matrice. O matrice *theta* conţine informaţii de structură, parametrii (estimaţi) pe structura dată şi acurateţea lor (de asemenea estimată).

Pentru modelele structurilor cu ieşiri multiple si cu evidenţierea spaţiului stărilor există reprezentarea *thss*, analogă în bună măsură cu modul de reprezentare *theta*.

Comanda *help thss* lămureşte proprietăţile matricilor *thss* şi diferenţele dintre cele două reprezentări din aceeaşi familie. Nu sunt de ignorat elementele de datare calendaristică şi orară conţinute în aceste reprezentări, utile în identificarea în sens strict a unor rezultate stocate în aceste forme.

Matricile *theta* au în vedere structura de model intrare-ieşire polinomială foarte generală:

$$A(q)y(t) = [B(q) / F(q)]\, u(t-nk) + [C(q) / D(q)]\, e(t) \qquad (1.3)$$

cu *A, B, C, D* si *F* polinoame în operatorul de întârziere *q*, de ordinul n_a, n_b, n_c, n_d, d_f, respectiv.

Matricile *theta* sunt create de comenzile *poly2th, pem, iv, iv4, arx, armax, oe, bj, ar, ivar*. Ele pot fi transformate în alte reprezentări cu comenzile *trf, zp, th2poly, th2th, th2ss si th2par*.

Matricile *thss* sunt structuri care definesc în general modele liniare în spaţiul stărilor.

Sunt folosite de comenzile/funcţiile *pem, th2arx, th2par,th2ss, idsim, present, thinit, fixpar, unfixpar, th2th,* sunt create prin comenzile *ms2th, mf2th, arx, iv4, arx2th* şi pot fi modificate prin comenzile *pem, thinit, fixpar* şi *unfixpar*.

Detalierea conţinutului matricilor *theta* se poate obţine cu ajutorul funcţiei/comenzii *present*.

Apelul la comanda *help* urmată de un nume din listă clarifică acţiunea fiecărei funcţii din cele menţionate.

2.2. Proceduri din subpachetul *System Identification Toolbox*

Forma generală a modelelor intrare-ieşire pentru sistemele cu ieşire unică este

$$A(q)y(t) = \sum_{i=1}^{n} \frac{B_1(q)}{F_1(q)} u_1(t - n_k) + \frac{C(q)}{D(q)} e(t) \qquad (1.4)$$

cu u_t intrarea numărului *i* din cele n_u intrări, cu *y* ieşirea, cu *A*, B_t, *C*, *D* şi F_t polinoame în operatorul de deplasare q^{-1} (sau z^{-1}).

Structura generală este definită prin întârzierile n_{ki} şi prin ordinele polinoamelor implicate care coincid cu numărul de poli şi de zerouri ale modelului dinamic de la intrări la ieşire şi ale modelului perturbaţiilor de la *e* la *y*.

Câteva cazuri speciale, particulare, ale modelului de mai sus sunt urmatoarele:

ARX: $A(q)y(t) = B(q)\, u(t-n_k)+e(t)$
ARMAX: $A(q)y(t) = B(q)\, u(t-n_k)+C(q)\, e(t)$

$$OE: \quad y(t) = \frac{B(q)}{F(q)} u(t-n_k) + e(t) \qquad (1.5)$$

$$BJ: \quad y(t) = \frac{B(q)}{F(q)} u(t-n_k) + \frac{C(q)}{D(q)} e(t) \quad \text{(Box-Jenkins)}$$

Procedura ***arx*** este utilizată pentru identificarea sistemelor de tip *ARX*. Calculele se fac pe baza unor date (y,u) observate experimental.

Secvenţa care urmează execută un exemplu de identificare pe date „experimentale" generate prin simularea unui sistem dinamic de formă cunoscută.

```
clear
sisc=tf{[1 –1],[1 2 5]}   % se creează un sistem continuu în timp
step (sisc)               % se reprezintă grafic răspunsul la treaptă
unitară
sisd=c2d (sisc,0.01)      % se converteşte sistemul la timp discret
hold on                   % pe acelaşi grafică
step (sisd)               % se reprezintă răspunsul la treaptă unitara
[y,t]= step (sisd);       % se evaluează răspunsul la treaptă unitară
u1=sign ( randn (size (t) ) ); % se creeaza o intrare binar- aleatoare
 [y1,t]=lsim (sisd, u1, t);  % se evaluează răspunsul prin simulare
z=[y1 u1];                % se creează matricea cu coloanele y1,u1
thd=arx (z, [2 2 1])      % se aplică procedura de identificare pe z,
                          pe  structura cunoscută [na,nb,nk]=[2 2 1]
e=pe(z,thd);              % se calculează erorile de reprezentare
figure                    % pe un grafic nou
h2=gcf;                   % cu handler-ul h2
set(h2,'Position',[150 90 560 420]); % după poziţionarea graficului
plot(t,e)                 % se reprezintă erorile de modelare
y1=y1+0.0001*randn(size(t)); % se alterează ieşirile observate
z1=[y1 u1];               % se creează noua matrice z sub numele de
z1
thd1=arx(z1,[2 2 1])      % se aplică din nou procedura de identificare
e1=pe(z1,thd1);           % se calculează erorile de reprezentare
title ('Erori de modelare')
ylabel ('Date curate')
xlabel ('Timp (s) ')
subplot(2,1,2)
```

```
plot(t,e1)                    % se reprezintă erorile de modelare
ylabel ('Date cu erori')
xlabel ('Timp (s) ')
```

Procedura ***armax*** este utilizată pentru identificarea sistemelor de tipul *ARMAX*. O simplă înlocuire în secvenţa de mai sus a apelului la funcţia *arx* prin apelul la funcţia *armax*, înlocuire insotită de modificarea argumentului relativ la structură, [2 2 0 1] în loc de [2 2 1], face secvenţa utilizabilă pentru ilustrarea identificării unui sistem *ARMAX***.**

Procedura ***oe*** este utilizată pentru identificarea sistemelor modelate prin relaţii de tipul *OE*. Şi de data aceasta, o simplă înlocuire în secvenţa de mai sus a apelului la funcţia *arx* prin apelul la funcţia *oe* face secvenţa utilizabilă identificării unui sistem *OE*. Deşi semnificaţia componentelor vectorului de structură se modifică, acesta poate rămâne neschimbat: *[2 2 1]*.

Pentru modele de tipul *BJ* (*Box-Jenkins*) este utilizată procedura *bj*. Şi de această dată, în secvenţa de mai sus a apelului la funcţia arx prin apelul la funcţia bj face secvenţa utilizabilă pentru ilustrarea identificării unui sistem *BJ*. Vectorul de structură se modifică din *[2 2 1] in [2 0 0 2 1]*.

Invocarea comenzii *help* urmată de numele funcţiei particulare utilizate lămureşte semnificaţia vectorului de structură pentru fiecare caz în parte.

În secvenţa ***MATLAB*** de mai sus apare şi funcţia/comanda *pe*. Această comandă, în varianta de apelare

```
e = pe (z, th);
```

produce diferenţele (erorile) dintre datele experimentale ***z*** şi modelul *th* dat în forma *theta*. O invocare cu o lista mai completă de restituiri

```
[ e, v, w, a, b, c, d, f] = pe (z, th)
```

generează nu numai erorile de modelare ci şi alte informaţii asupra modelului. În *a, b, c, d* şi *f* sunt depuse polinoamele care apar în forma generală a modelului, iar în ceilalţi doi vectori cantităţile

w=(b/f) u si
v = a [y -w].

O comandă/ funcţie care poate face tot ce pot face funcţiile referite mai devreme este funcţia *pem,* cu următoarele posibilitati de apelare:

th=pem (z, thstruc);
th=pem (z, thstruc, index);

şi cu alte posibilităţi sondabile prin comanda *help pem*. În aproape toate cazurile, *thstruc* este o matrice cu structura modelului şi cu o parametrizare, fie ea si provizorie, în genul celor generate de funcţia *poly2th*. Se poate însă apela funcţia *pem* şi cu *thstrc* în formă vectorială, simplă

[na nb nc nd nf nk].

Forma de apelare fără *index* face operaţia de identificare cu iniţializarea liberă sau inspirată de parametrii din matricea *thstruc,* dacă aceasta este completă.

Forma cu argumentul *index* face aceeaşi operaţie, dar numai pentru parametrii din linia specificată prin acel index. Se înţelege că, în acest din urmă caz, matricea *thstruc* trebuie să fie completă.

În ambele cazuri, introducerea unui ultim argument de tip şir cu valoarea *'trace'* produce afişarea pe monitor a etapelor calculului de estimare.

3. MODUL DE LUCRU

- Dacă nu este deja creat, se creează un director/folder de lucru.
- Se activează platforma **MATLAB** şi se introduce comanda

Cd (calea spre directorul/folderul de lucru)

- Se apelează funcţiile care creează sisteme cu obiecte de toate tipurile, continue sau discrete in timp. Se memorează sistemele şi apoi se recuperează, după o comandă intermediară *clear* (care eliberează memoria din spaţiul **MATLAB** de lucru curent). Se verifică conţinutul spaţiului de lucru curent cu comenzile *who* şi/sau *whos*.
- Din imaginaţie sau din lucrările anterioare se creează modele de tipuri variate ale unor sisteme dinamice. Se aplică sistemelor intrarea *impulse* sau intrarea *step*, prin comenzile adecvate descrise sumar mai sus, si se urmăresc răspunsurile sistemelor.
- Se explorează conversiile posibile între diferite forme de modele şi între diferite forme sintetice de stocare a caracteristicilor lor.
- Se extrag caracteristicile numerice ale răspunsurilor - lista de valori ale ieşirilor şi vectorul momentelor la care răspunsurile sunt observate.
- Se creează un *script* după modelul dat în secţiunea ***Breviar teoretic***. Scriptul va fi executat repetat cu schimbările semnalate în acea secţiune.
- Se explorează identificarea în condiţii de zgomot variate. Exemplul dat introduce un zgomot cu abaterea medie pătratică de ***0,0001***, coeficientul numeric din linia

*y1=y1+0.0001*randn (size (t));*

- Se modifică structura propusă şi se repetă operaţia de estimare de parametri.
- Din imaginaţie sau din lucrări anterioare se creează sisteme dinamice şi modele de structuri diverse. Se introduc în *script* şi se face succesiv identificarea lor.
- Se supun observării critice în identificare şi legătura lor cu structurile alternative propuse şi cu acurateţea datelor.

LUCRAREA 2

SIMULAREA COMPORTĂRII SISTEMELOR LINIARE DE ORDINUL I ŞI ORDINUL II FOLOSIND PACHETUL DE PROGRAME *MATLAB*

1. OBIECTIVELE LUCRĂRII

Studiul comportării sistemelor liniare de ordinul I şi ordinul II la intrări uzuale (treaptă, rampă, sinusoidă, aleatoare).

2. BREVIAR TEORETIC

Funcţia de transfer a sistemului de ordinul I simulat este:

$$H(s)=\frac{k}{Ts+1}\,, \tag{2.1}$$

iar funcţia de transfer a sistemului de ordinul II :

$$H(s)=\frac{kw_n^2\left(\tau s+1\right)}{s^2+2\zeta w_n s+w_n^2} \tag{2.2}$$

cu notaţiile uzuale: k pentru factorul de amplificare, w_n pentru pulsaţia naturală, ζ pentru factorul de amortizare . Cazul $\tau=0$ este deplin reprezentativ, dar şi cazurile cu gradul în s al numărătorului egal cu unitatea sunt de luat în considerare.

Pentru a realiza o simulare cât mai sugestivă a unui sistem este necesar a fi precizate câteva elemente cum sunt:

- Orizontul de timp pe care se realizează simularea;
- Numărul de puncte în care se cere a fi evaluat răspunsul sistemului;
- Intervalul de timp între două momente succesive, care coincide uzual cu un interval de eşantionare.

Toate aceste elemente depind de particularităţile sistemului (tip, constantele sistemului) precum şi de tipul semnalului aplicat la intrarea sistemului.

Sistem de ordinul I
O alegere posibilă a constantelor este *k=2*, *T=5 s* .

Secvenţa **MATLAB**

```
num=2;          % introducerea numărătorului
den=[5 1];      % şi a numitorului funcţiei H(s)
step(num,den);  % simularea şi reprezentarea grafică a
                  răspunsului
```

corespunde cazului în care programul **MATLAB** alege automat, după reguli proprii, elementele de timp. Pentru a putea aduce la vedere aceste elemente ultima instrucţiune se execută sub forma:

```
[y,x,t]=step(num,den);  % alegerea elementelor temporale şi
                          simularea.
```

In continuare

```
plot(t,y);          % reprezentarea grafică a răspunsului
length(t);          % afişarea lungimii vectorului t
t(2)-t(1);          % afişarea intervalului dintre două evaluări
t(length(t));   % afişarea orizontului de timp
```

Forma obişnuită a vectorului momentelor *t* este $t=0:p:t_{max}$, cu *p* distanţa dintre două puncte succesive şi cu t_{max} orizontul de timp pe care se face simularea .

Se deduce uşor că numărul de elemente ale vectorului *t* aste de $(t_{max}/p)+1$. Timpul pe care se observă simularea pentru un sistem de ordinul I se ia de obicei de apx. patru ori coeficientul *T* din expresia funcţiei de transfer, pentru a cuprinde şi o zonă din regimul aproape staţionar.

Iată o secvenţă **MATLAB** cu alegerea interactivă a vectorului temporal:

```
t=0:0.5:25;            % iniţializarea vectorului t
y=step(num,den,t);     % calcule de simulare
plot(t,y);             % reprezentarea grafică a răspunsului
```

Intrarea sistemului poate fi şi de alte tipuri diferite de forma treaptă. În toate aceste cazuri se utilizează funcţia **MATLAB** *lsim*.

Următoarea secvenţă simulează sistemul şi răspunsul lui la un semnal rampă:

```
t=0:0.5:25;            % initializarea vectorului t
u=t;                   % realizarea semnalului rampă
y=lsim(num,den,u,t);   % calcule de simulare
plot(t,[y u']);        % reprezentarea grafică a intrarii şi a răspunsului
```

Dacă intrarea este sinusoidală, este necesar a se alege pasul de eşantionare mai mic de un sfert din perioada sinusoidei.

Secvenţa urmatoare de instrucţiuni simulează răspunsul sistemului de ordinul I la intrarea sinusoidală.

```
a=1;                   % amplitudinea sinusoidei
per=3;                 % perioada semnalului sinusoidal
w=2*pi/per;            % pulsaţia (frecvenţa unghiulară)
t=0:per/16:5*per;      % iniţializarea vectorului temporal

u=a*sin(w*t);          % evaluarea eşantioanelor intrării
y=lsim(num,den,u,t);   % calculul de simulare
plot(t,[y u']);        % reprezentarea grafică a intrării şi a
   răspunsului
```

Răspunsul tinde la o sinusoidă după ce depăşeşte regimul tranzitoriu.

Pentru o intrare aleatoare, secvenţa **MATLAB** corespunzătoare este prezentată în continuare:

```
dt=1;                    % intervalul de eşantionare
tm=25;                   % orizontul de timp pe care se face simularea
t=0:dt:tm;               % vectorul temporal
for i=1:length(t)
  u(i)=-1+rand(1);       % realizarea intrării aleatoare
end;
y=lsim(num,den,u,t);  % calculul de simualare
plot(t,[y  u']);                % reprezentarea grafică a intrării şi răspunsului
```

Sistem de ordinul II

Se propune mai întâi scrierea funcţiilor **MATLAB** pentru sistemele de ordinul II următoare:

```
function zvar(k,wn,nrz)
% zvar(k,wn,nrz)
% Calculează răspunsul la intrarea treaptă unitară al unei familii
% de sisteme de ordinul II cu amplificarea k si pulsaţia naturală wn,
% la un factor de amortizare variabil între 0 şi 1, nrz>1 este numărul
% de valori pentru factorul zeta sau numărul graficelor-răspuns obţinute
% Exemplu de apelare a funcţiei: zvar(1,1,10)
num=k*wn^2;
den=[1 0 wn^2];          % valoarea din den(2) se completează mai tarziu
tmax=20/wn;              % se stabileşte durata simulării
t=0:0.1:tmax;
z=zeros(length(t),nrz);  %matricea pentru răspuns
if nrz>1
  dz=1/(nrz-1);          % increment pentru zeta
   zeta=-dz;             % se iniţializează zeta
  for i=1:nrz
    zeta=zeta+dz;
    x(i)=zeta;
    den(2)=2*zeta*wn;
    z(:,i)=step(num,den,t);
   end
  mesh(t,x,z')
  xlabel('Timp')
```

```
  ylabel('Factor de amortizare','Position',[-1,0.3,0])
    zlabel('Iesire')
    title('Raspunsul la intrare treapta unitara pentru factor de
amortizare variabil')
    end

  function zfix(k,wn,zeta)
  % zfix(k,wn,zeta)
  % Calculează răspunsul la intrare treapta unitară al unui sistem
  % de ordinul II cu amplificarea k, pulsaţia naturală wn, şi factorul
  % de amortizare zeta cu reprezentare grafică
  % Exemplu de apelare a funcţiei: zfix(1,1,0.2)
  num=k*wn^2;                % coeficienţi numărător
  den=[1 2*zeta*wn wn^2];    % coeficienţi numitor
  tmax=20/wn;                % se stabileşte durata simulării
  t=0:0.1:tmax;              % vectorul timp al eşantionărilor
  z=step(num,den,t);         % matricea pentru răspuns
  plot(t,z)
  xlabel('Timp')
  ylabel('Iesire')
  sir=num2str(zeta);
  sir=strcat('Raspunsul la intrare treapta unitara pentru factorul de
amortizare zeta =',sir);
  title('Sir')
  hold on
  y=[1 1];
  x(1)=t(1);
  x(2)=tmax;
  grid
 plot(x,y)
  hold off

  function isin(k,wn,zeta)
  % isin(k,wn,zeta)
 % Calculează răspunsul la intrări sinusoidale al sistemului de ordin
II
  % cu amplitudinea k, pulsaţia naturală wn, şi factorul de amortizare
zeta
  % cu reprezentarea grafică şi corelarea cu diagrama Bode
```

```
% Exemplu de apelare a funcţiei: isin(1,1,0.4)
wi=[0.5 1.0 1.5];   % factori de modificare a frecvenţei la intrare
dt=0.1/wn;
tmax=500*dt;
t= 0:dt:tmax;
num=k*wn^2;
den=[1 2*zeta*wn^2];
clf
for i=1:3
  u=sin(wi(i)*wn*t);
  y(:,i)=lsim(num,den,u,t);
end
figure(1)
for i=1:3
  subplot(3,1,i)
  plot(t,y(:,i))
  axis([0,tmax,-5,5])
  grid on
  if i==3
    xlabel('Timp')
  end
  sir=num2str(wi(i));
  sir=strcat('Iesire la w=',sir,'*wn');
  ylabel(sir)
end
figure(2)
w=logspace(log10(wn)-1,log10(wn)+1);
m=bode(num,den,w);
clf
loglog(w,m)
grid on
xlabel('Pulsatia w')
ylabel('Amplificare')
title('Diagrama Bode')
set(gcf,'Position',[140 100 560 420])
```

Reprezentările grafice obţinute ca urmare a rulării secvenţelor de program exemplificate mai sus sunt prezentate în figura 2.1 şi în figura 2.2.

3. MODUL DE LUCRU

- Se activează platforma **MATLAB** şi se introduc succesiv secvenţele de program menţionate mai devreme.

Cerinţe pentru sistemele de ordinul I

- Pentru cazul aplicării semnalului de tip treaptă se modifică lungimea vectorului *t* de la 20 la cca. 100 puncte, se modifică orizontul de timp menţinând numărul de puncte la 101 puncte; se încearcă evaluarea coeficienţilor *k* şi *T* pe grafic; se modifică cele două valori şi se observă stabilitatea sistemului.
- În fiecare caz se examinează aspectele legate de reprezentarea grafică.
- Pentru intrarea treaptă se încearcă identificarea prin metoda indicială, se fac aprecieri asupra preciziei.
- Pentru intrarea sinusoidală se studiază regimul tranzitoriu şi tendinţa spre regimul staţionar.
- Se recomandă studiul atent al funcţiei de generare a numerelor aleatoare *rand* şi al comportării sistemului în cazul utilizării unei intrări aleatoare.

Cerinţe pentru sistemele de ordinul II

- Prin apelarea funcţiei *zvar* se obţine o reprezentare în trei dimensiuni pe care se poate observa influenţa factorului de amortizare asupra formei răspunsului la un salt treaptă unitar, al unui sistem de ordinul II cu poli complex conjugaţi. Valorile factorului de amortizare sunt în acest caz subunitare şi pozitive;
- Prin apelarea repetată a funcţiei *zfix*, se obţin grafice ale răspunsului unui sistem de ordinul II pentru valori fixe ale factorului de amortizare chiar supraunitare sau negative;
- În fiecare caz se examinează aspectele legate de reprezentarea grafică, aspectele privitoare la stabilitatea sistemului;
- Pe graficele pentru valori fixate ale factorului de amortizare se încearcă identificarea prin metoda indicială, se fac aprecieri asupra preciziei evaluărilor;

- Apelarea funcţiei *isin* permite observarea comportării unui sistem de ordinul II dacă la intrare se aplică o sinusoidă de amplitudine maximă unitară. Se pot face comparaţii cu diagrama Bode;
- Se recomandă realizarea unui grafic cu toate răspunsurile pe aceeaşi diagramă pentru facilitarea comparării amplitudinilor la ieşire în zona staţionară (regimul permanent);
- Se fac comparaţii cu diagrama Bode;
- În vederea altor aplicaţii, se vor studia funcţiile din biblioteca **MATLAB** utilizate în secvenţele de program din lucrare (*grid*, *logspace*, *clf*, *loglog*, *plot* etc.).

Fig.2.1. Reprezentări grafice pentru sistemele de ordinul I

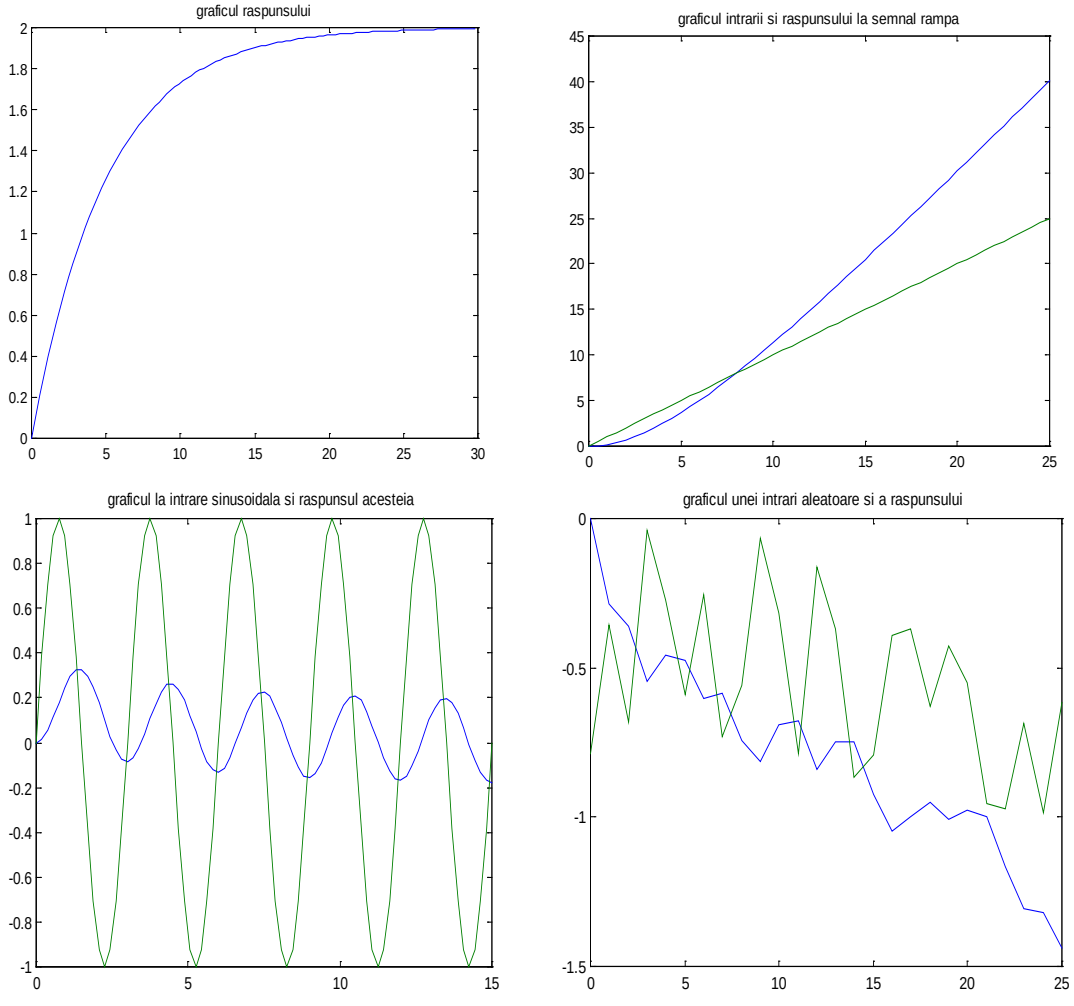
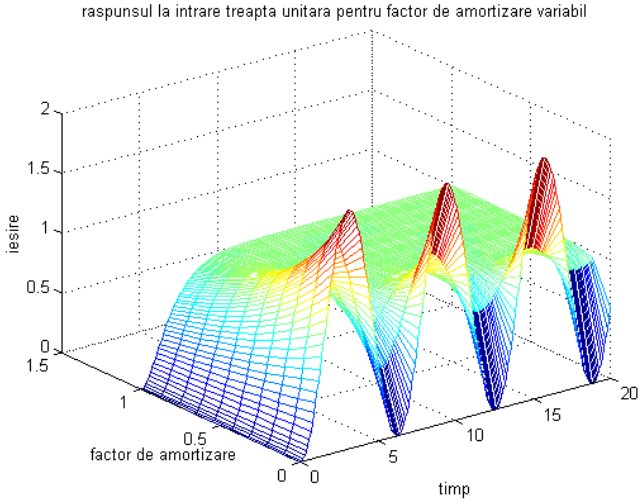
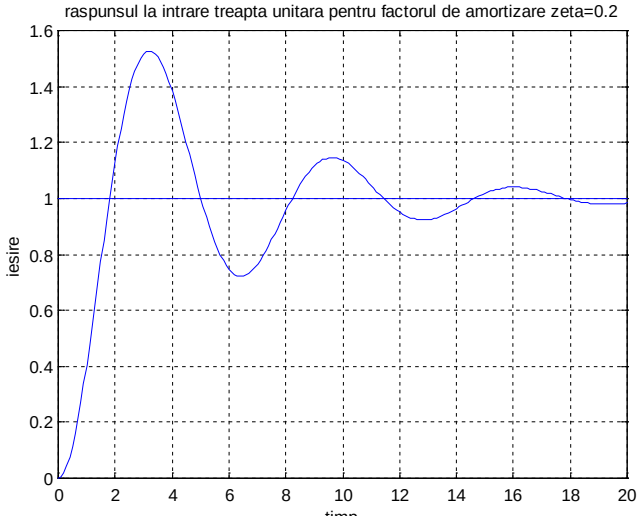
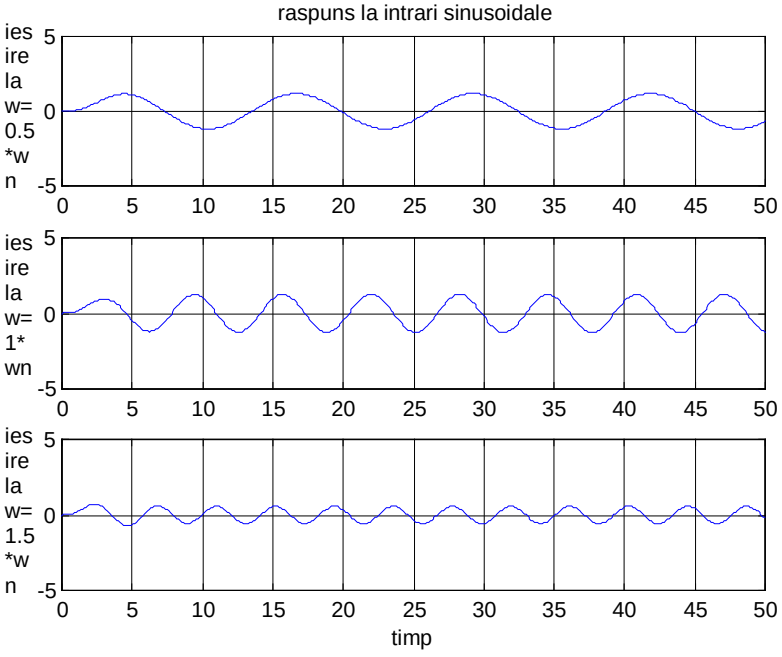


Fig.2.2 Reprezentări grafice pentru sistemele de ordinul II







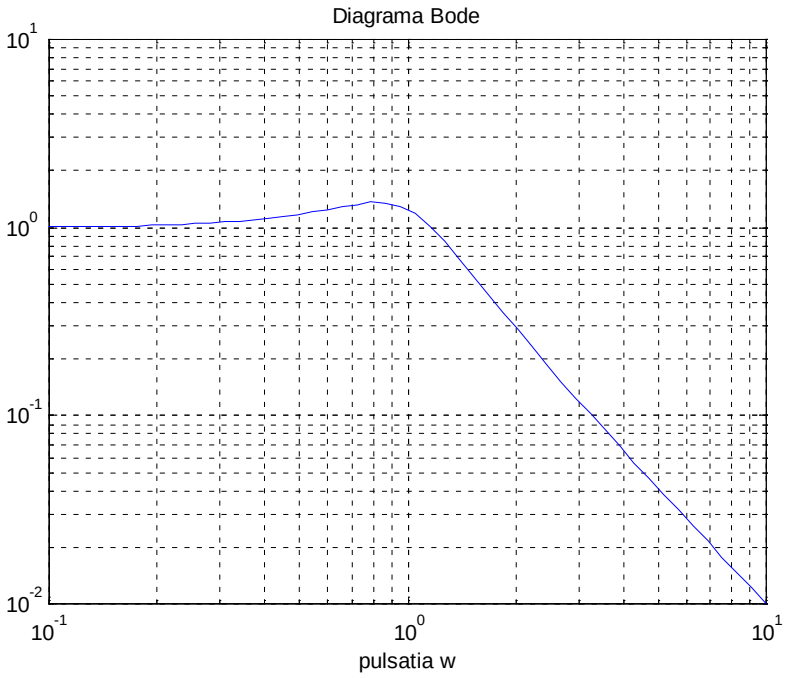


Diagrama Bode
10^1
10^0
10^-1
10^-2
10^-1
10^0
10^1
pulsatia w

LUCRAREA 3

ALGORITMI DE TIPUL FFT (*FAST FOURIER TRANSFORM*). CALCULUL FUNCŢIILOR DE CONVOLUŢIE ŞI COVARIANŢĂ FOLOSIND FFT

1. OBIECTIVELE LUCRĂRII

Studiul transformatei Fourier folosind algoritmul FFT implementat cu pachetul de programe **MATLAB**. Calculul functiilor de convolutie si covarianta folosind FFT.

2. BREVIAR TEORETIC

2.1. Algoritmul FFT

Este esenţial, pentru început, să fie subliniat faptul că algoritmul **FFT** pentru calculul transformatei Fourier discrete a unei secvenţe este ″punctul nodal″ al procesării semnalelor digitale. El este aplicat în filtrare, convoluţie, calculul răspunsului la frecvenţă, ca şi în aplicaţii referitoare la estimarea spectrului de putere.

fft(x) este transformata Fourier discretă a vectorului x, calculată cu o transformată Fourier rapidă. Dacă *X* este o matrice, ***fft(X)*** este transfornmata Fourier rapidă a fiecărei coloane a lui *X*.

fft(x,n) este transformata Fourier rapidă în *n* puncte. Dacă lungimea lui *x* este mai mică decât *n*, *x* este completat cu zerouri până la lungimea lui *n*. Dacă lungimea lui *x* este mai mare decât *n*, secvenţa *x* este trunchiată. Când *X* este matrice, lungimea coloanelor este ajustată în acelaşi fel.

ifft(x) este transformata Fourier inversă a vectorului *x*.

Cele două funcţii care urmează implementează perechea transformata Fourier – transformata Fourier inversă dată:

$$X(k+1)=\sum_{n=0}^{N-1} x(n+1)W_N^{kn}$$

$$X(n+1)=\frac{1}{N}\sum_{k=0}^{N-1} X(k+1)W_N^{-kn}, \qquad (3.1)$$

unde $W_N = e^{-j(2\pi/n)}$ şi N=lungime(x)

Exemplu

FFT a unui vector coloană *x*, de forma

x=[4 3 7 –9 1 0 0 0]

se găseşte cu

y=fft(x)

şi dă drept rezultat

```
y=6.0000
   11.4853      -2.7574i
    -2.0000     -12.0000i
```

$$h=\frac{fft(b,h)}{fft(a,n)} \qquad (3.2)$$

2.2. Funcţii de convoluţie şi de corelaţie

Există o colecţie de funcţii disponibile pentru convoluţie, deconvoluţie şi pentru calculul estimărilor funcţiilor de corelaţie, şi anume:

conv – funcţii de convoluţie
deconv – funcţii de deconvoluţie
xcorr – funcţii de transcorelare
xcov – funcţii de transcovarianţă
corrcoef – coeficienţi de corelaţie
cov – matrice de covarianţă

2.2.1. Convoluţie şi deconvoluţie

Propoziţia

c=conv(a,b)

face convoluţia vectorilor a şi b.

Suma de convoluţie este

$$c(n+1)=\sum_{k=0}^{N-1} a(k+1)b(n-k) \tag{3.3}$$

unde N este lungimea secvenţei maxime.

Operaţia

[q,r]=deconv(b,a)

face deconvoluţia vectorului a din vectorul b. Rezultatul este întors în vectorul q şi restul în vectorul r, astfel încât *b=conv(q,a)+r*

Dacă a,b sunt vectori cu coeficienţi polinomiali, convoluţia este echivalentă cu multiplicarea celor două polinoame, iar deconvoluţia este o împărţire polinomială.

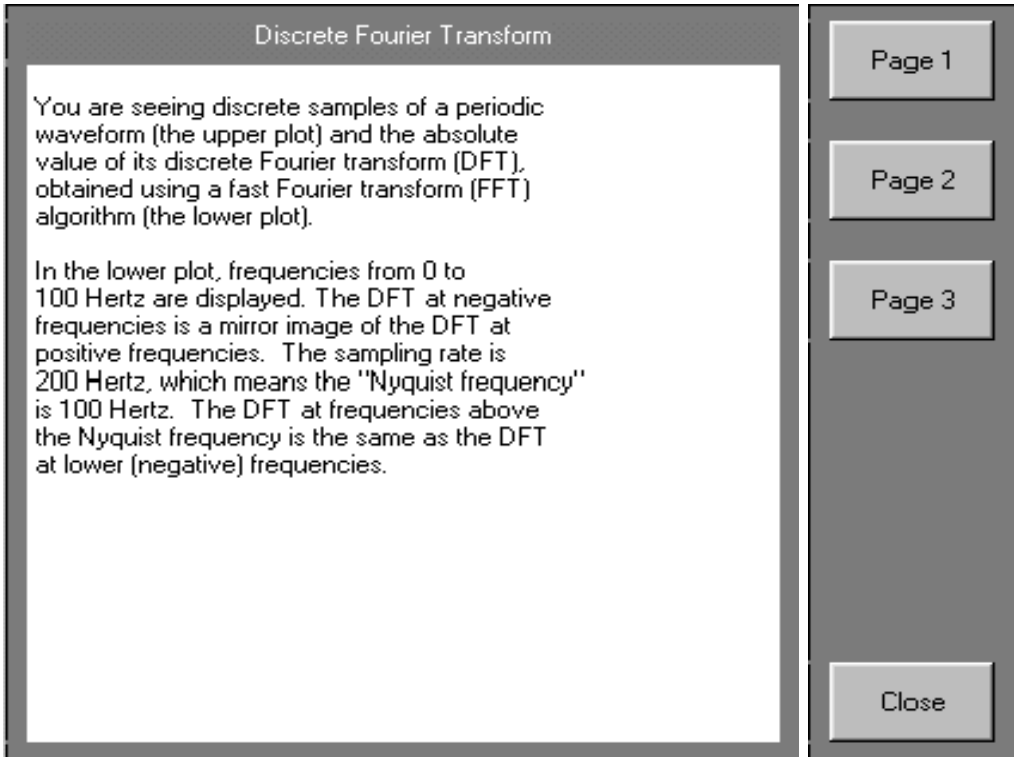
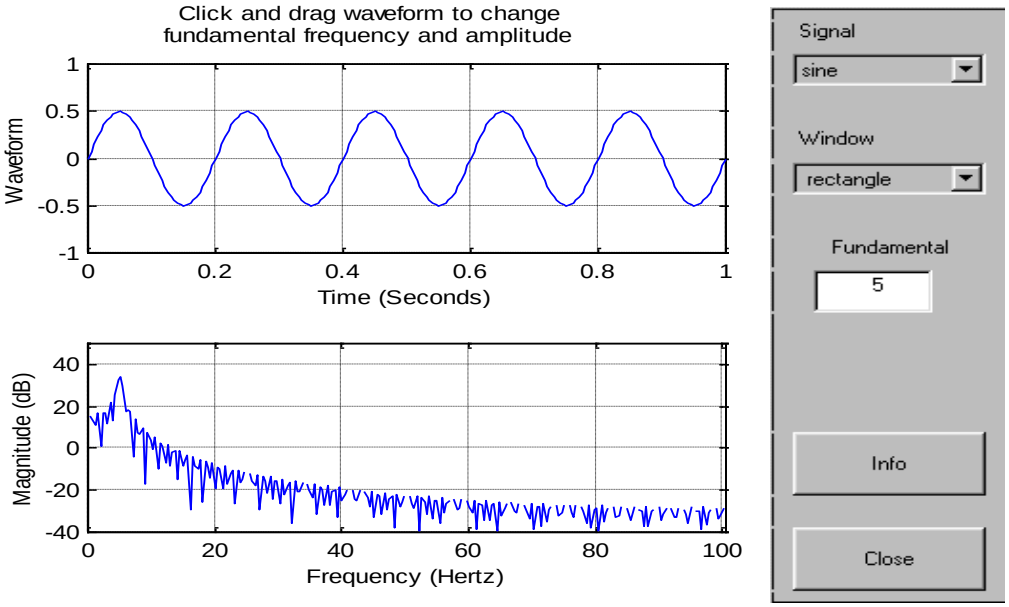
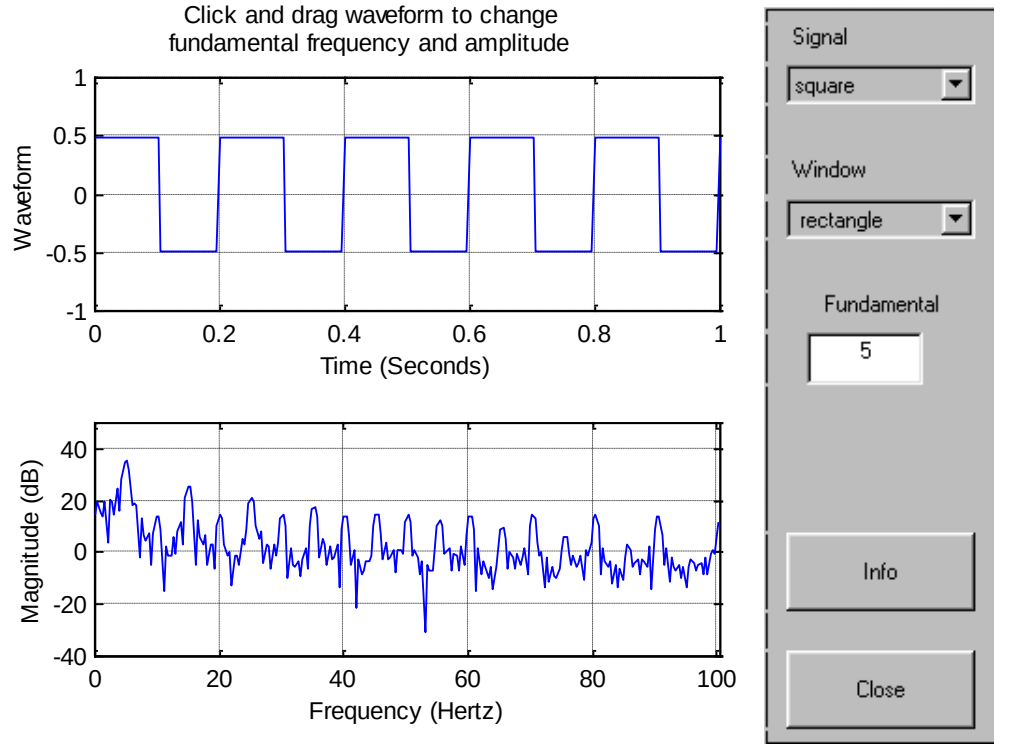


Fig. 3.1. Fereastra de prezentare în MATLAB a transformatei Fourier discretă


Click and drag waveform to change
fundamental frequency and amplitude
Waveform
Time (Seconds)
Magnitude (dB)
Frequency (Hertz)
Signal
sine
Window
rectangle
Fundamental
5
Info
Close


Click and drag waveform to change
fundamental frequency and amplitude
Waveform
Time (Seconds)
Magnitude (dB)
Frequency (Hertz)
Signal
square
Window
rectangle
Fundamental
5
Info
Close

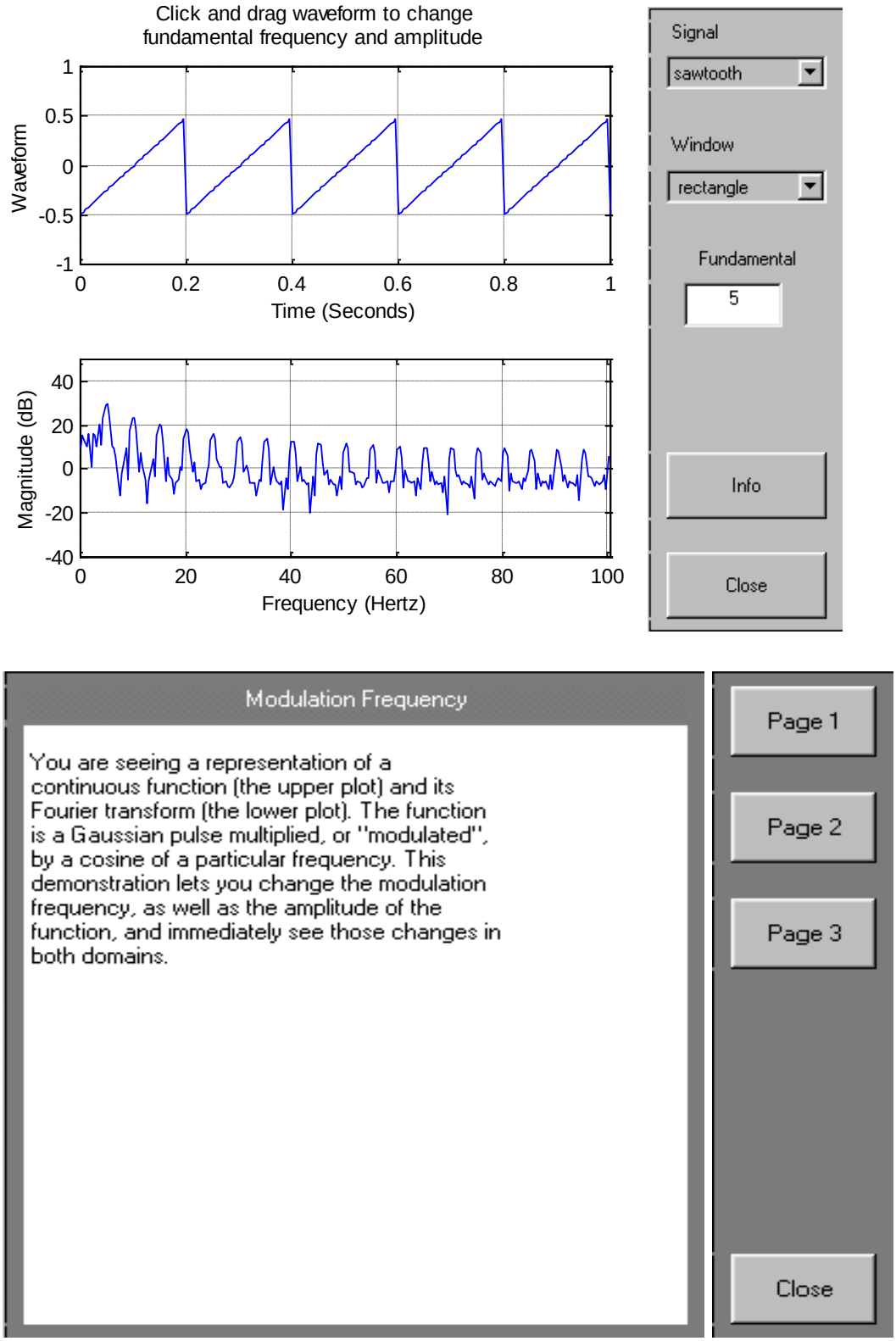
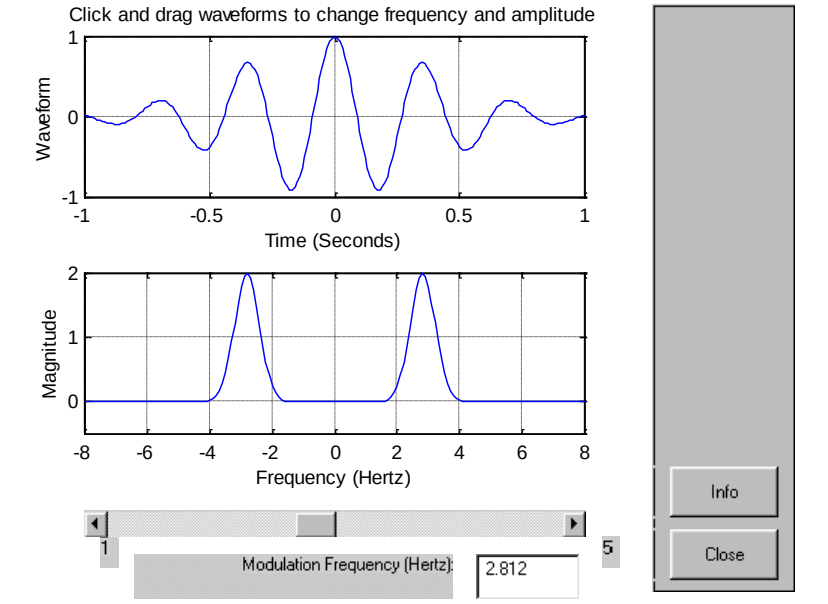
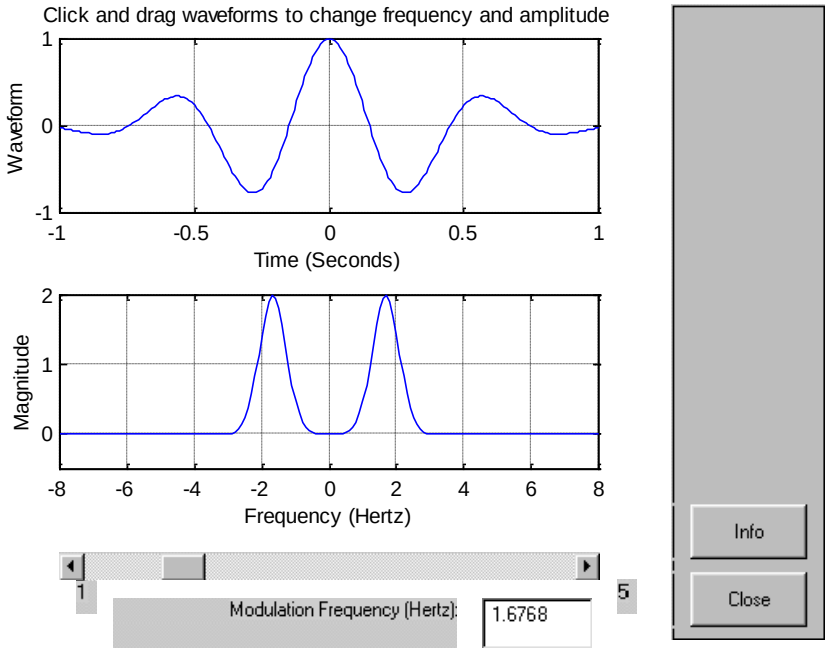


Fig.3.2. Fereastra de prezentare in MATLAB a transformatei Fourier continue





3. MODUL DE LUCRU. PROBLEME PROPUSE

- Calculul FFT a vectorului coloană v = [9 8 12 –4 6 5 5 5].

  fprintf('Transformata Fourier a vectorului:')
  v=[9 8 12 -4 6 5 5 5]
  fprintf('este :')
  tfv=fft(v)

- Calculul FFT a matricei A=[1 2 3, 4 5 6, 7 8 9],

```
fprintf('Transformata Fourier a matricei:')
a=[1 2 3 ; 4 5 6 ;7 8 9]
fprintf('este :')
tfa=fft(a)
```

- Calculul FFT inversă a matricei B=[6 7 8 9, 4 5 6 8, 1 1 2 3, 4 7 6 7].

```
fprintf('Transformata Fourier inversa a matricei:')
b=[6 7 8  9 ;4 5 6 8 ;1 1 2 3 ;4 7 6 7]
fprintf('este :')
itfb=ifft(b)
```

- Cu setul de date obţinut dintr-un cosinus eşantionat cu 10 paşi pe perioadă: y=cos(2πk/10), interpolaţi datele cu un pas dublu şi verificaţi valorile obţinute.

Se interpolează datele cu o singură variabilă utilizând metoda FFT (Fast Fourier Transform) cu ajutorul funcţiei ***interpft.***

- Considerând funcţiile *x=5+6t*, *y=6t* şi *z=5sin(t)*, se cere calculul coeficienţilor de corelaţie Rxy şi Rxz, pentru domeniul t Є [0,5].

.

```
t=0:1:5;
x=5+6*t;
y=6*t;
z=5*sin(t);
plot(t,x,t,y,t,z);
rxy=corrcoeff(x,y)
rxz=corrcoeff(x,z)
```

- Cu funcţiile de la punctul precedent, dar cu *t* de forma

$t=3w^2+3\cos(2\pi w/3)$,

calculaţi coeficienţii de corelaţie Rxy şi Rxz, pentru w Є [0,25].

- Ce concluzii trageţi în urma efectuării punctelor de mai sus unde aţi folosit funcţia *corrcoef* ?

Ce dependenţă există între x şi y, dar între x şi z? Justificaţi răspunsul.

• Pentru matricile de la punctele de mai sus, să se afle matricea de covarianţă.
Indicaţie : se va folosi funcţia ***cov****.*

• Considerând A = [5 6 2 1] şi B = [4 3 2 1] doi vectori care conţin coeficienţii a două polinoame să se facă convoluţia şi deconvoluţia acestora.
Indicaţie : se va folosi funcţiile ***conv*** *şi* ***deconv****.*

• Folosind funcţiile de transcorelare şi de transcovarianţă, (***xcorr*** şi respectiv ***xcov***), cu un set de date ales arbitrar, să se interpreteze rezultatele obţinute.

LUCRAREA 4

INDICATORI STATISTICI AI MĂSURĂRILOR. FUNCŢII MATLAB PENTRU CALCULE STATISTICE

1. OBIECTIVELE LUCRĂRII

Studiul funcţiilor **MATLAB** pentru calcule statistice. Aplicaţii.

2. BREVIAR TEORETIC

Funcţiile uzuale **MATLAB** folosite pentru calcule statistice sunt următoarele:

1) cumsum(x,dim) – calculează suma cumulată a elementelor vectorului *x* de dimensiune *dim*.
Pentru o matrice rezultatul este tot o matrice cu dimensiunile lui x şi conţine suma cumulată pentru fiecare coloană.

Exemplu

Pentru X=[1 2 3; 1 2 3],

cumsum(X,1);

are ca rezultat matricea [1 2 3;2 4 6]

iar *cumsum(X,2);*

are ca rezultat vectorul [1 3 6;1 3 6];

2) cumprod(x,dim) – calculează produsul cumulat al elementelor vectorului *x*. Pentru *x* matrice rezultatul este tot o matrice cu dimensiunile lui *x* şi conţine produsul cumulat al fiecărei coloane.

Exemplu

cumprod(x,1)

are ca rezultat matricea [1 2 3; 1 4 9]

cumprod(x,2)

are ca rezultat matricea [1 2 6; 1 2 6].

3) corrcoef(x) – calculează o matrice pentru coeficienţi de corelaţie pentru un vector x, în care fiecare linie a matricei este o observaţie, iar fiecare linie este o variabilă.

corrcoef(x,y,) , unde x şi y sunt vectori coloană este acelaşi lucru cu a scrie ***corrcoef([x,y])***.

Coeficienţii de corelaţie ai datelor se folosesc pentru a stabili dacă între două seturi de date înregistrate în 2 vectori diferiţi există o dependenţă liniară.

Exemplu

x=[1 2 3; 2 7 5; 3 5 8]
a=[1 2 3;4 5 6; 7 8 9];
r=corrcoef(x)

are ca rezultat
r=

1.0000	*0.5960*	*0.9934*
0.5960	*1.0000*	*0.5000*
0.9934	*0.5000*	*1.0000*

iar
r=corrcoef(a,x)

are ca rezultat

r=

1.0000 0.7614
0.7614 1.0000

4) cov(x) – dacă x este un vector funcţia întoarce varianţa acestuia. Dacă x este o matrice cu liniile observaţii şi coloanele variabile aceasta funcţie va returna o matrice de covarianţă

Exemplu

cov(x) pentru matricea de mai sus va returna

1.0000 1.5000 2.5000
1.5000 6.3333 3.1667
2.5000 3.1667 6.3333

5) diff(x) – diferenţa dintre numerele succesive.
Pentru un vector cu elementele $x_1 \ldots x_n$ diferenţa este tot un vector calculat astfel $[x_2 - x_1 \ldots x_n - x_{n-1}]$, iar pentru o matrice se face diferenţa dintre liniile succesive.

Exemplu

a= [2 4 3 5 1 7];
diff(a) = [2 -1 2 -4 6]
x=[1 2 3; 2 7 5; 3 5 8];
diff(x) = [1 5 2;1 -2 3]

6)n=hist(y) – împarte elementele lui y în 10 intervale egale şi returnează numărul de elemente din fiecare interval. Dacă y este o matrice *hist* va lucra în josul coloanelor.

Hist() – fără argumente produce o histogramă, conform exemplelor prezemtate in figura 4.1 şi figura 4.2.

Exemplul 1

Secvenţa de program

a= [2 4 3 5 1 7]
hist(a)

produce reprezentarea grafică din figura 4.1.

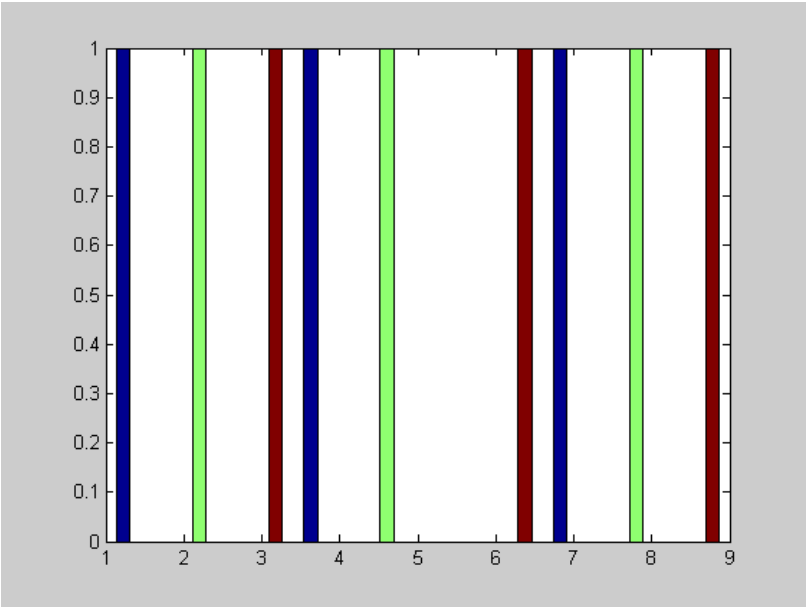


Fig.4.1. Histograma corespunzătoare exemplului 1

Exemplul2

Secvenţa de program

x=[1 2 3; 2 7 5; 3 5 8];
hist(x).

produce histograma din figura 4.2.

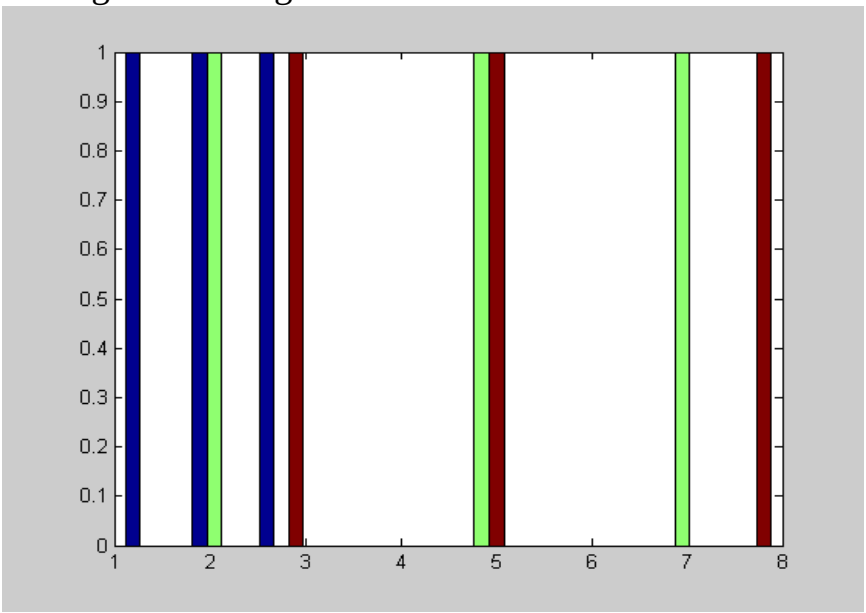


Fig.4.2. Histograma corespunzătoare exemplului 2

7) max(x), min(x) – returnează valoare maximă, respectiv minimă a componentelor vectorului x.
Dacă x este matrice returnează într-un vector maximul/minimul de pe fiecare coloană.

Exemplu

a= [2 4 3 5 1 7] ;
min(a)= 1
x=[1 2 3; 2 7 5; 3 5 8];
max(x)=[3 7 8]

8) mean(x) – returnează valoarea medie a unui set de date dintr-un vector x. Dacă datele sunt elementele unei matrice, valoarea medie este conţinută de un vector care are elementele valorile medii ale fiecărei coloane.

Exemplu

a= [2 4 3 5 1 7] ;
x=[1 2 3; 2 7 5; 3 5 8];
mean(a)=3,667
mean(x)=[2.0000 4.6667 5.3333]

9) prod(x) – calculează produsul elementelor unui vector, iar pentru o matrice rezultatul este un vector care are ca elementele produsul de pe fiecare coloană a matricei.

Exemplu

a= [2 4 3 5 1 7];
x=[1 2 3; 2 7 5; 3 5 8];
prod(a)=840
prod(x)=[6 70 120]

10)sort(x) – sortează elementele unui vector sau matrice în ordine crescătoare (la matrice sortarea se face pe fiecare coloană)

Exemplu

a= [2 4 3 5 1 7] ;
x=[1 2 3; 2 7 5; 3 5 8];

sort(a)=[1 2 3 4 5 7]
sort(x)=[1 2 3; 2 5 5; 3 7 8]

11) std(x) – calculează abaterea standard.

Exemplu

a= [2 4 3 5 1 7] ;
x=[1 2 3; 2 7 5; 3 5 8];
std(a)= 2.1602
std(x)=[1.0000 2.5166 2.5166]

12) sum(x) – calculează suma elementelor unui vector, iar pentru o matrice se obţine un vector cu elemente ce au valoarea egală cu suma de pe fiecare coloană.

Exemplu

a= [2 4 3 5 1 7] ;
x=[1 2 3; 2 7 5; 3 5 8];
sum(a)=22
sum(x)=[6 14 16]

13) trapz(x) – calculează integrala folosind metoda trapezelor luând în considerare şi spaţiile.

Exemplu

a= [2 4 3 5 1 7] ;
x=[1 2 3; 2 7 5; 3 5 8];
trapz(a)= 17,5
trapz(x)=[4 10,5 10,5]

14) table1(tab,x0) – returnează un tabel cu interpolarele liniare ale liniilor din tabelul *tab*.

Exemplu

a= [2 4 3 ; 5 1 7] ;
r=table1(a,3)= [3 4,333]

15) y=interpft(x,n) – returnează vectorul y cu lungimea n obţinut prin interpolarea lui x prin metoda transformatei Fourier.

Dacă x este o matrice interpolarea se face pe fiecare coloană.

Exemplu

```
a= [2    4    3 ; 5    1    7] ;
y=interpft(a,2)= [2    4    3
                  5    1    7]
```

16) polyfit(x,y,n) – aproximează un set de date cu un polinom P(x) de gradul n.

Exemplu

```
a=[2 4 3;5 1 7];
b=[1 4 2; 1 5 7];
polyfit(a,b,3)
ans =[-0.0109    0.5471   -3.3582    7.192]
```

17) griddata(x,y,z,xi,yi) – interpolează prin metoda distanţei inverse valoarea unei funcţii de două variabile x, y.

3. MODUL DE LUCRU

3.1. Probleme rezolvate

1. Fie doi vectori x=[-2 -1 0 2 4]; y=[-15 -3 2 3 10]; Să se aplice o procedura de regresie liniară celor doi vectori şi să se reprezinte grafic rezultatul obţinut.

Rezolvare

```
x=[-2 -1 0 2 4 ];
y=[-15 -3 2 3 10];
coef=polyfit(x,y,1);
xn=-2:1:4;
y1=polyval(coef,xn)
plot(xn,y1,'r');
coef = 3.4828   -2.6897
y1 =-9.6552   -6.1724   -2.6897    0.7931    4.2759    7.7586
      11.2414
```

Reprezentarea grafică a dreptei obţinute în urma aplicării procedurii de regresie liniară este cea din figura 4.3.

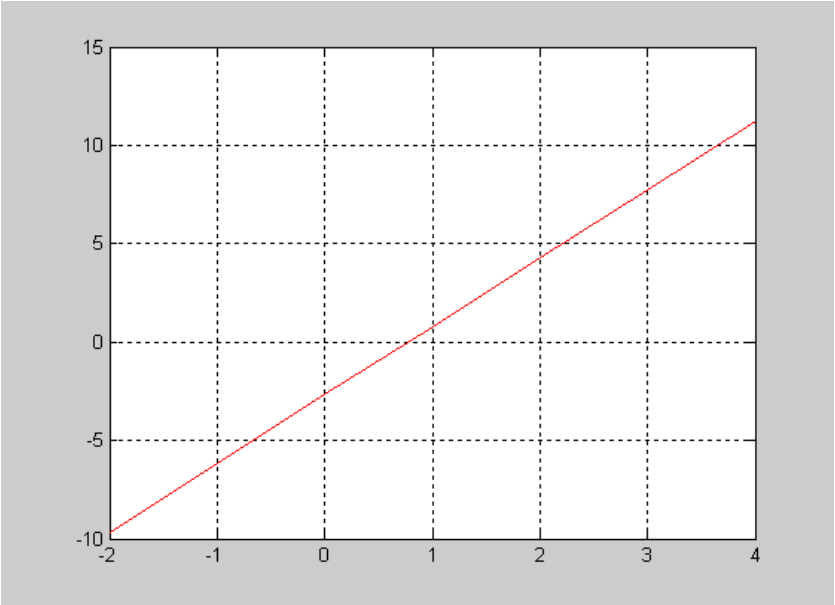


Fig.4.3. Regresie liniară

2. Pentru un servomotor cu poziţioner s-au obţinut datele experimentale din tabelul următor:

P[bar]	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
H[mm]	0	5	12	16	21	27	32	32	32

Să se traseze caracteristica statică a servomotorului cu poziţioner .

Rezolvare

```
p=[0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1];
h1=[0 5 12 16 21 27 32 32 32];
table1(p,0.2);
plot(p,h1);
grid on;
ylabel('H[mm]');
xlabel('presiunea [bar]');
title('CARACTERISTICA STATICA A SERVOMOTORULUI
      CU POZITIONER');
```

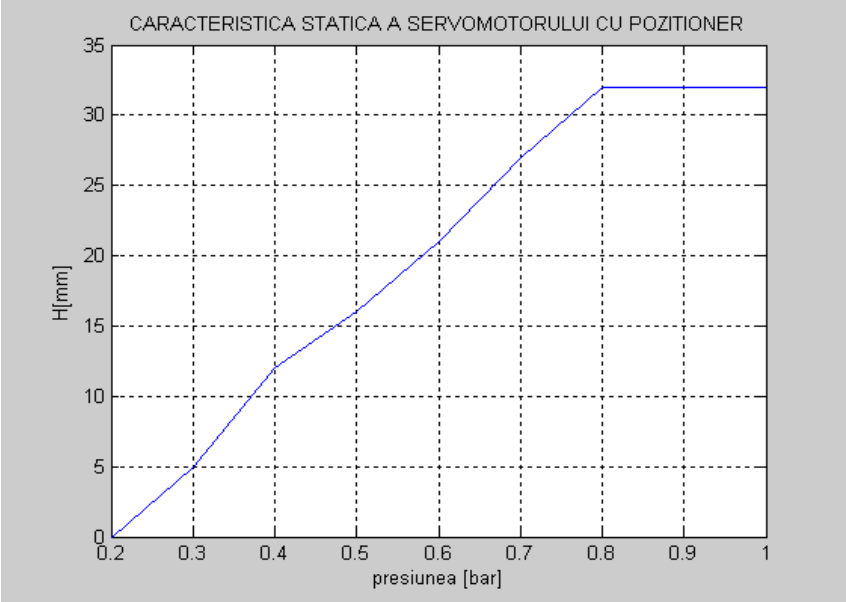


Fig.4.4. Caracteristica statică a servomotorului cu poziţioner

3.2. Probleme propuse

1. Să se determine minimul şi maximul vectorului V şi matricei M, unde :

V=[1 3 -9 0]
M=[1 2 3; -4 0 9; 13 7 -10]

2 Să se sorteze matricele A şi B de forma:

A=[1 -2 3; 5 -9 0 ; -10 3 0]
B=[7 2 -5 4 -1; 5 8 1 -6 -4; 2 0 -3 6 9]

3. Să se realizeze prin interpolare graficul unei funcţii ştiind că acesta conţine punctele A(1,1), B(2,3), C(2.5,5), D(4,6), E(6,10).

4. Fie vectorii

x=[-2 -1 0 2 4]; y=[-15 -3 2 3 10].

Să se aplice o procedură de regresie liniară celor doi vectori.

5. Fie matricea A, de forma

A=[1 3 2 5; 3 5 7 6; 2 8 5 9; 4 2 1 1].

Să se calculeze:

- suma cumulată;
- produsul cumulat;
- produsul şi suma elementelor;
- să se sorteze mai întâi după linii şi apoi după coloane.

Să se reprezinte grafic histograma corespunzătoare.

LUCRAREA 5

FILTRAREA DATELOR EXPERIMENTALE AFECTATE DE ZGOMOTE

1. OBIECTIVELE LUCRĂRII

Studiul posibilităţilor de diminuare a zgomotelor care însoţesc datele experimentale recoltate în scopul identificării unui sistem.

2. BREVIAR TEORETIC

Datele experimentale sunt însoţite de cele mai multe ori de zgomote. Figurile 5.1.si 5.2. ilustrează răspunsul normalizat al unui sistem liniar de ordinul I la intrare treaptă unitară neafectat, respectiv afectat de prezenţa zgomotului.

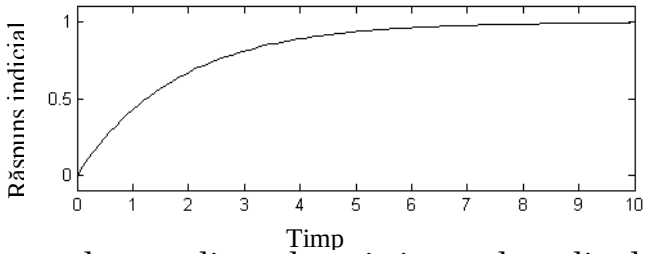


Fig. 5.1. Răspunsul normalizat al unui sistem de ordinul I la intrare treaptă unitară, neafectat de zgomote.

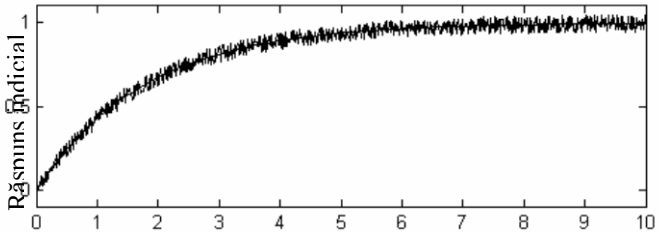


Fig. 5.2. Răspunsul normalizat al unui sistem de ordinul I la intrare treaptă unitară, afectat de zgomote.

Pentru identificarea sistemului prin metoda indicială este necesară evaluarea derivatei într-un moment oarecare, în particular în origine, şi apoi intersectarea dreptei tangente la curbă cu dreapta *y=1*. Diferenţa absciselor punctului în care se evaluează derivata şi a celui de intersecţie a tangentei cu orizontala regimului staţionar este exact constanta de timp τ a sistemului. Evaluări ale derivatelor şi a funcţiei răspuns se cer şi pentru alte sisteme de ordine variate. Dacă evaluările pot fi acceptate în cazul datelor puţin (sau deloc) afectate de zgomot, pentru situaţiile când zgomotul devine important, evaluările pe calea clasică, în esenţă o cale grafică, sunt imposibile.

O metodă de a netezi curbele experimentale de genul celor prezentate anterior este filtrarea printr-un filtru *trece-jos*. Este vizibil faptul că fenomenul tranzitoriu este relativ lent, aşadar este reprezentat de frecvenţele joase ale spectrului, în timp ce fluctuaţiile datorate zgomotului sunt fenomene considerabil mai rapide, legate prin urmare de frecvenţe mai ridicate din spectru. Pentru semnale analogice se pot utiliza filtre *trece-jos* analogice. Un simplu divizor *RC* cu ieşirea pe capacitate, de exemplu, este un filtru *trece-jos*, fără calităţi deosebite, dar este un filtru care atenuează cu precădere frecvenţele înalte.

Datele recoltate uzual sunt date numerice de tip eşantioane prelevate cu o anumită regularitate ale unor funcţii definite pe intervale de timp dense pe axa reală a timpului. Există o varietate de filtre numerice de tipul *trece-jos* care pot fi utilizate pentru diminuarea zgomotului.

Filtrele numerice cele mai simple sunt *filtrele de mediere*. Ieşirea unui astfel de filtru se prezintă ca media aritmetică a unui număr de eşantioane succesive

$$y(t) = \frac{1}{n}\sum_{i=1}^{n} x(t-i+1) \qquad (5.1)$$

Graficele care urmează ilustrează efectul filtrării prin medierea a trei sau a cinci valori consecutive.

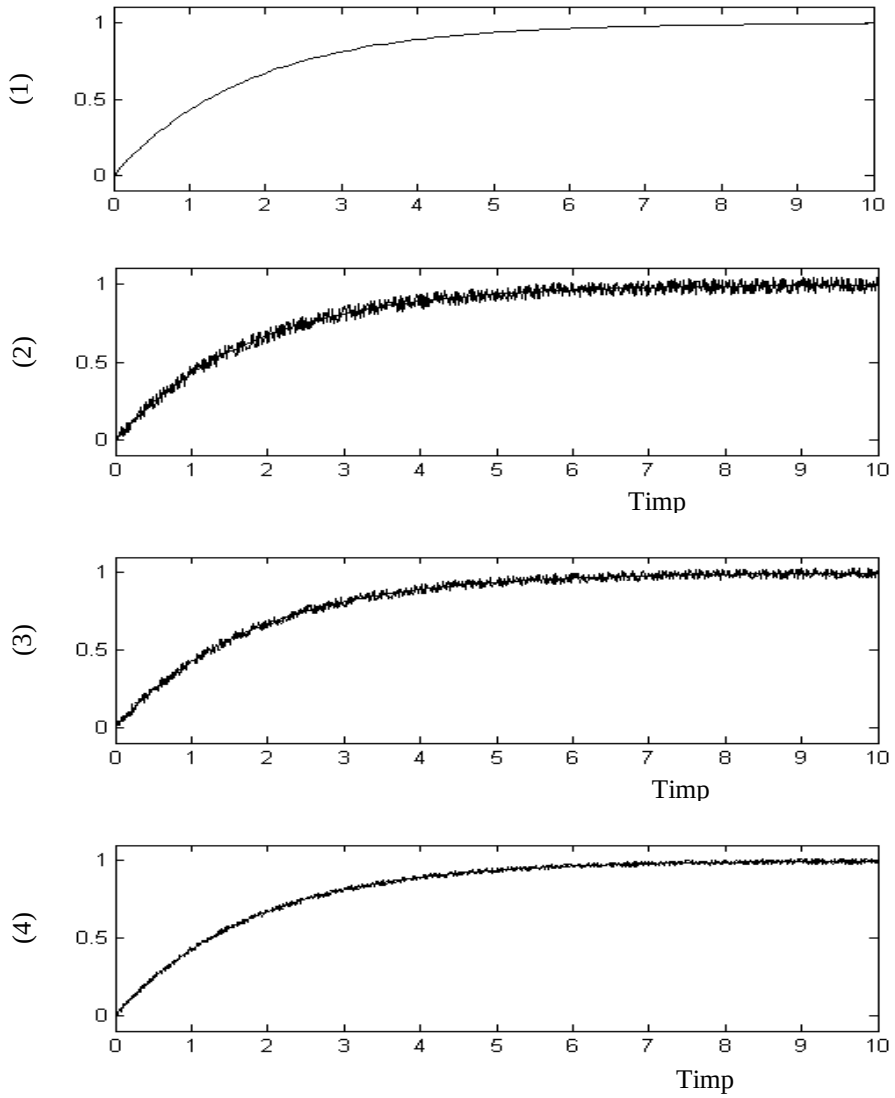


Fig.5. 3. Efectul filtrării prin medierea valorilor consecutive

Se observă evoluţia calitativă a datelor pe măsură ce numărul de eşantioane mediate creşte: trei pentru figura (3), cinci pentru figura (4). Datele brute, nefiltrate sunt reprezentate în figura (2). Se reia pentru ilustrare şi graficul (1) cu răspunsul neafectat de zgomot.

Graficele au fost generate cu secvenţa **MATLAB** următoare:

```
% Programul realizează filtrarea unui semnal afectat de zgomot
%Sintaxa de apelare este Filtrare(k,a,m)
%k-coeficientul din functia de transfer
%a-timpul de intarziere al sistemului
%m-numarul de date luate in fiecare medie facuta

Function Filtrare(k,a,m)
sis=tf(k,[a,1];
[y,t]=step(sis);
hold on;
z=rand(size(y));
y1=y+(z-5)*.05;
pause;
plot(t,y1);
for j=1:109-m;
    s=0;
    forI=j:j+m;
        s=s+y1(I);
        end
   y2(j)=1/(m+1)*s;
end
hold off;
pause;
plot(t,y2,'g');
pause;
```

Filtrarea prin mediere nu este singura modalitate de a obţine din date afectate de zgomot, date mai puţin marcate de fluctuaţiile satistice. Un efect de filtrare *trece-jos* are şi *interpolarea răspunsului din eşantioanele sale prin mijlocirea funcţiei sinc*

$$sinc(x)=\begin{cases}1 & x=0\\ \dfrac{\sin(\pi x)}{\pi x} & x\neq 0\end{cases} \qquad (5.2)$$

combinată cu o fereastră finită.

Se cunoaşte faptul că un semnal de bandă limitată se poate reconstitui pe baza relaţiei:

$$s(t) = \sum_{n=-\infty}^{+\infty} s\left(\frac{n}{2W}\right) \frac{\sin 2\pi W\left(t - \frac{n}{2W}\right)}{2\pi W\left(t - \frac{n}{2W}\right)} \tag{5.3}$$

dacă frecvenţa eşantioanelor este cel puţin egală cu dublul frecvenţei celei mai mari din spectrul de frecvenţe al semnalului. Relaţia de mai sus, cunoscută şi sub numele de *teorema eşantionării*, care conţine amintita funcţie *sinc,* este nepractică din cauza sumei infinite pe care o conţine. Se recurge, de aceea, la un număr finit de eşantioane ceea ce se poate interpreta ca "privirea" eşantioanelor printr-o *fereastră rectangulară,* descrisă de relaţia

$$w(t) = \begin{cases} 1, & t \in \left(-\frac{\alpha}{2}, \frac{\alpha}{2}\right) \\ 0, & \text{în rest} \end{cases} \tag{5.4}$$

şi care este de lărgime finită α, definită în jurul originii dar centrată succesiv pe momente de timp variate, momente în care se urmăreşte a fi (re)evaluat semnalul *s(t)* din eşantioanele sale. Relaţia de interpolare se modifică după cum urmează

$$s(t) = \sum_{n=-\infty}^{+\infty} s\left(\frac{n}{2W}\right) w\left(t - \frac{n}{2W}\right) \frac{\sin 2\pi W\left(t - \frac{n}{2W}\right)}{2\pi W\left(t - \frac{n}{2W}\right)} \tag{5.5}$$

şi cu toate că valorile extreme ale indicelui după care se face însumarea au fost menţinute aceleaşi, infinite, însumarea se face de fapt pe un număr finit de eşantioane, cele cuprinse în fereastră. Formula de interpolare cu fereastră dată mai sus se utilizează şi pentru semnale de bandă nelimitate (semnalul observat la ieşirea unui sistem când la intrarea lui se aplică o variaţie treaptă este de această natură) cu pierderi de componente din partea superioară a spectrului. De aici utilul şi aşteptatul efect de filtrare/diminuare a zgomotelor însoţitoare ale unui semnal.

Utilizarea ferestrei rectangulare este în toate privinţele echivalentă cu filtrarea prin mediere dacă eşantioanele sunt ponderate astfel ca suma ponderilor să fie egală cu unitatea. Stabilirea acestor ponderi se face simplu prin normalizarea eşantioanelor unui semnal cuprinse în fereastră.

Fereastra rectangulară nu este singurul tip de fereastră utilizat în calcule de filtrare. Există posibilitatea de a alege între mai multe tipuri de ferestre şi de a selecta o deschidere anume pentru fereastra reţinută pentru calcule. Sunt date imediat câteva ferestre dintre cele mai utilizate.

Fereastra **Bartlett**

$$Bartlett(x,\tau)=\begin{cases}1-\dfrac{|x|}{\tau} & \text{pentru} \quad |x|<\tau \\ 0 & \text{în rest}\end{cases} \tag{5.6}$$

Fereastra **Blackmann**

$$Blackmann(x,\tau)=\begin{cases}0.42+0.50\cdot\cos\left(\pi\dfrac{x}{\tau}\right)+0.08\cdot\cos\left(2\pi\dfrac{x}{\tau}\right) & |x|<\tau \\ 0 & \text{în rest}\end{cases} \tag{5.7}$$

Fereastra **Gauss**

$$Gauss(x,\tau,\sigma)=\begin{cases}2^{-\left(\frac{x}{\sigma}\right)^2} & |x|<\tau \\ 0 & \text{în rest}\end{cases} \tag{5.8}$$

Ferestrele **Hann** şi **Hamming**, foarte asemănătoare, diferite numai prin parametrul α (α=0.5, respectiv α=0.54)

$$H(x,\tau,\alpha)=\begin{cases}\alpha+(1-\alpha)\cos\left(\pi\dfrac{x}{\tau}\right) & |x|<\tau \\ 0 & \text{în rest}\end{cases} \tag{5.9}$$

Fereastra **Kaiser** cu parametrul ajustabil α care controlează cât de rapid se apropie de zero laturile ferestrei

$$Kaiser(x,\tau,\alpha)=\begin{cases}\dfrac{I_0(\alpha\sqrt{1-(x/\tau)^2})}{I_0(\alpha)} & |x|<\tau \\ 0 & \text{în rest}\end{cases} \tag{5.10}$$

cu $I_0(x)$ funcţia Bessel modificată de ordinul zero.

Fereastra **Lanczos**, lobul central al funcţiei *sinc* extinsă la un interval dat

$$Kaiser(x,\tau,\alpha)=\begin{cases}\dfrac{\sin\left(\pi\dfrac{x}{\tau}\right)}{\pi\dfrac{x}{\tau}} & |x|<\tau \\ 0 & \text{în rest}\end{cases} \tag{5.11}$$

Fereastra **Parzen**, o aproximare cubică pe porţiuni a ferestrei Gauss de întindere doi.

$$Parzen(x,\tau,\alpha)=\begin{cases}(2+x)^3 & \\ 4-6x^2-3x^3 & -1\le x\le 0 \\ 4-6x^2+3x^3 & 0\le x\le 1 \\ (2-x)^3 & 1\le x\le 2 \\ 0 & \text{în rest}\end{cases} \tag{5.12}$$

Fereastra **Welch,** descrisă de relaţia (5.13)

$$Welch(x,\tau,\sigma)=\begin{cases}1-\left(\dfrac{x}{\tau}\right)^2 & \text{pentru} \quad |x|<\tau \\ 0 & \text{în rest}\end{cases} \tag{5.13}$$

In figura 5.4. este reprezentat grafic un semnal sinusoidal, iar în figurile 5.5. şi 5.6, acelaşi semnal sinusoidal filtrat utilizând, respectiv, ferestrele Bartlett şi Kaiser.
Graficele au fost generate cu următoarea secvenţă de program :

```
function grafice(tau,alfa)
t=1;
for x=-tau-1:.1:tau+1;
   y(t)=sin(sin(10*x)/7+cos(9.5*x/2)/10)/2;
   t=t+1;
end
x=-tau-1:.1:tau+1;
figure
qq=polyfit(x,y,3);
pp=polyval(x,qq);
xx=-tau-1:.001:tau+1;
nnv=spline(x,y,xx);
plot(xx,nnv);
title('Semnal  sinusoidal
sin(sin(10*x)/7+cos(9.5*x/2)/10)/2');
hold on;
grid;
t=1;
figure;
for x=-tau-1:.1:tau+1;
   if ((x>=-tau)&(x<=tau))
       bartlett(t)=1-abs(x)/tau;
   else
      bartlett(t)=0;
   end
   t=t+1;
end
x=-tau-1:.1:tau+1;
qq=polyfit(x,bartlett,3);
pp=polyval(x,qq);
xx=-tau-1:.001:tau+1;
nnv=spline(x,bartlett,xx);
plot(xx,nnv,'r');
%title('Fereastra Bartlett');
hold on;
t=1;
yt1=y+bartlett;
qq=polyfit(x,yt1,3);
pp=polyval(x,qq);
xx=-tau-1:.001:tau+1;
nnv=spline(x,yt1,xx);
plot(xx,nnv);
title('Filtrarea cu ajutorul ferestrei Bartlett')
grid;
t=1;
figure;
```

```
for x=-tau-1:.1:tau+1;
   if ((x>=-tau)&(x<=tau))
      kaiser(t)=besseli(0,x)*(alfa*(1-
(x/tau)^2)^(1/2)/besseli(0,alfa));
   else
      kaiser(t)=0;
   end
   t=t+1;
end
x=-tau-1:.1:tau+1;
qq=polyfit(x,kaiser,3);
pp=polyval(x,qq);
xx=-tau-1:.001:tau+1;
nnv=spline(x,kaiser,xx);
plot(xx,nnv,'r');
hold on;
%title('Fereastra Kaiser');
t=1;
%figure
yt2=y+kaiser;
qq=polyfit(x,yt2,3);
pp=polyval(x,qq);
xx=-tau-1:.001:tau+1;
nnv=spline(x,yt2,xx);
plot(xx,nnv);
title('Filtrarea cu ajutorul ferestrei Kaiser')
grid;
```

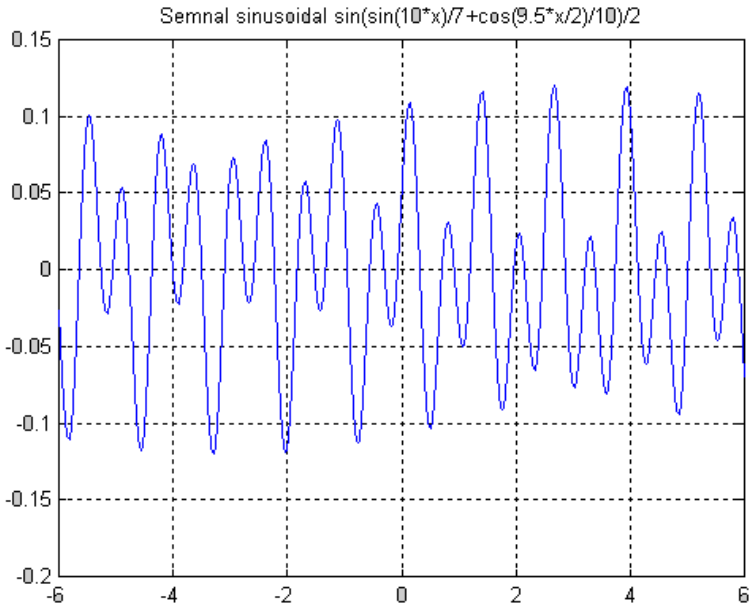


Fig.5.4. Semnal sinusoidal nefiltrat

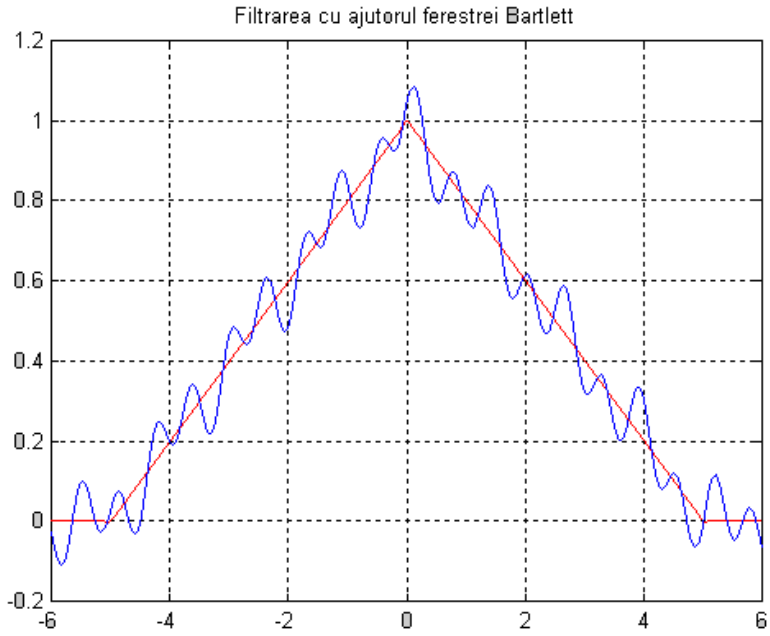


Fig.5.5. Semnal sinusoidal filtrat cu fereastra Bartlett

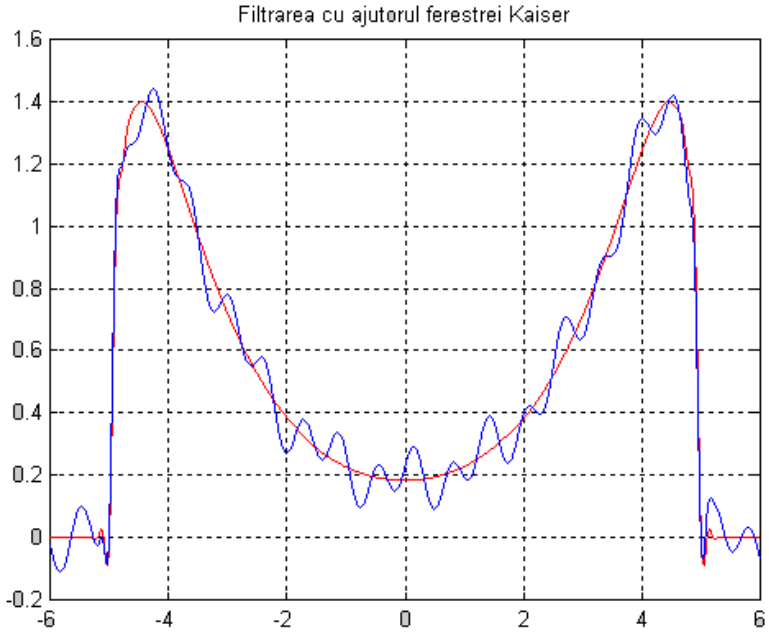


Fig.5.6. Semnal sinusoidal filtrat cu fereastra Kaiser

În identificarea prin metoda indicială se utilizează derivarea curbei răspunsului la intrarea treaptă. Pentru evaluarea derivatei există posibilitatea utilizării funcţiei *cosc*, de forma

$$\cos c(x) = \begin{cases} \dfrac{\cos(\pi x)}{x} - \dfrac{\sin(\pi x)}{\pi x^2} & \text{pentru} \quad x \neq 0 \\ 0 & \text{pentru} \quad x = 0 \end{cases} \qquad (5.14)$$

care este derivata funcţiei *sinc* menţionată mai devreme. Derivarea termen cu termen a sumei din teorema eşantionării produce o sumă de funcţii *cosc* cu coeficienţi proporţionali cu eşantioanele semnalului. Prin utilizarea unor ferestre din cele prezentate mai sus se pot obţine estimări ale derivatei semnalului.

3. MODUL DE LUCRU

- Se generează o secvenţă de eşantioane ale răspunsului indicial al unui alt sistem liniar de ordinul I şi se observă efectul zgomotului şi efectul filtrării *trece-jos* prin mediere.

- Se elaborează programe care definesc funcţii capabile să filtreze un semnal prin mijlocirea celorlalte ferestre din cele prezentate în **Breviar teoretic –** Blackmann, Gauss, Hann, Hamming, Parzen, Lanczos. Se observă efectul filtrării *trece-jos* realizate.

- Se elaborează programul de implementare al unui algoritm de calcul al derivatei cu funcţia *cosc* şi fereastră rectangulară. Se observă efectul de filtrare la evaluarea derivatei pe baza datelor-eşantioane.

- Se evaluează derivata răspunsului cu funcţia *cosc* şi fereastra rectangulară în condiţii variate de zgomot.

LUCRAREA 6

ESTIMATORI M. ESTIMATORI DE PARAMETRI ROBUŞTI

1. OBIECTIVELE LUCRĂRII

Studiul comportării aşa-numiţilor *estimatori M*, estimatori de parametri înrudiţi cu binecunoscutul estimator bazat pe metoda celor mai mici pătrate, dar cu robusteţe îmbunătăţită în prezenţa erorilor grosiere prezente în date.

2. BREVIAR TEORETIC

O tehnică robustă foarte populară în estimarea de parametri este tehnica aşa-numiţilor *estimatori M.*

Fie r_i rezidualul experimental *i*, adică diferenţa dintre observaţia *i* şi valoarea dată de un model. Metoda standard a celor mai mici pătrate încearcă să minimizeze suma pătratelor acestor reziduale $\Sigma_i \; r_i^2$. Încercarea poate fi instabilă dacă în date sunt prezente erorile grosiere (*outliers*). Datele cu erori grosiere pot produce un efect atât de important în minimizarea sumei pătratelor încât parametri estimaţi să fie grav distorsionaţi.

Estimatorii M pot reduce efectul erorilor grosiere prin înlocuirea rezidualelor luate la pătrat din expresia clasică a celor mai mici pătrate cu o altă funcţie de reziduale, $\Sigma_i \rho(r_i)$, cu ρ o funcţie simetrică (pară) pozitiv definită, cu un minim unic în origine şi cu o creştere mai lentă decât pătratul.

În locul rezolvării directe a problemei se implementează o metodă iterativă a celor mai mici pătrate reponderate după algoritmul prezentat în continuare.

Fie $p=[p_1 \ \dots \ p_m]^T$ vectorul parametrilor modelului care trebuie estimaţi din date experimentale. Produsul unui estimator M care utilizează funcţia ρ este vectorul p soluţie a ecuaţiilor

$$\Sigma_i \psi(r_i)\partial r_i / \partial p_j = 0, \qquad \text{pentru } j=1,2,\dots,m \tag{6.1}$$

în care funcţia (derivată) $\psi(x)=d\rho(x)/dx$ este numită *funcţie de influenţă*.

Prin definirea *funcţiei pondere*

$$w(x)=\frac{\psi(x)}{x} \tag{6.2}$$

ecuaţia de mai sus devine

$$\Sigma_i w(r_i)\, r_i * \partial r_i / \partial p_j = 0 \qquad \text{pentru } j=1,2,\dots,m \tag{6.3}$$

ceea ce este exact sistemul de ecuaţii care se obţine la rezolvarea problemei iterative a celor mai mici pătrate reponderate

$$min \Sigma_i w(r_I^{(k-1)})\, r_i^2 \tag{6.4}$$

cu indicele superior ataşat iteraţiei curente. Ponderile trebuie evaluate după fiecare iteraţie pentru a fi utilizate în iteraţia următoare.

Funcţia de influenţă $\psi(x)$, după cum şi numele indică, este o măsură a influenţei pe care o observaţie o are asupra parametrilor estimaţi. Pentru cele mai mici pătrate, de exemplu, $\rho(x)=x^2/2$ şi funcţia de de influenţă este $\psi(x)'x$, adică influenţa unei singure observaţii, nu importă care, asupra estimaţiei parametrilor creşte liniar cu mărimea erorii, ceea ce reprezintă un semn al lipsei de robusteţe a metodei.

Un estimator este ***robust*** dacă influenţa unei singure observaţii este insuficientă pentru a produce o deplasare semnificativă în estimaţii. Un estimator *M* robust trebuie să satisfacă un număr de condiţii:

- Să aibă o funcţie de influenţă mărginită;
- Să asigure unicitatea soluţiei. Funcţia obiectiv prin minimizarea căreia se obţine vectorul de parametri *p* trebuie să fie unimodală (cu un singur extrem) ceea ce matematic corespunde unei convexităţi a funcţiei ρ în variabila *p*;
- Ori de câte ori derivata a doua $\partial^2\rho(.)/\partial p^2$ este singulară, funcţia obiectiv trebuie să aibă un gradient nenul, $\partial\rho(.)/\partial p \neq 0$, pentru a evita căutarea prin întreg spaţiul parametrilor.

Câteva dintre funcţiile uzual utilizate în estimarea de parametri sunt prezentate în Tabelul 6.1. , cu proprietăţile:

- Estimatorul L_2 (al celor mai mici pătrate) nu este robust din cauză că funcţia de influenţă nu este mărginită.
- Estimatorul L_1 (valoarea absolută) nu este stabil deoarece funcţia $\rho=|x|$ nu este strict convexă şi derivata a doua în origine nu este mărginită. O soluţie nedeterminată este oricând posibilă. Estimatorul L_1 reduce influenţa erorilor mari dar acestea au încă influenţă din cauză că funcţia de influenţă nu are puncte de tăiere.
- Estimatorul L_1–L_2 împrumută de la estimatorul L_1 capacitatea de a reduce influenţa erorilor mari şi de la estimatorul L_2, proprietatea de a fi convex.
- Funcţiile din L_P (ale celor mai mici puteri) alcătuiesc o familie de funcţii. Cu $\nu=2$ se regăseşte estimatorul L_2, cu $\nu=1$ cazul se reduce la L_1. Cu cât ν este mai redus cu atât incinta erorilor mari asupra estmaţiilor *p* este mai lipsită de importanţă. Se pare că un ν moderat asigură un estimator relativ robust, slab perturbat de erorile grosiere. Investigaţiile unor cercetători duc la valori ale lui ν în apropiere de valoarea *1,2* sunt însă probleme de calcul: rezidualele nule fac calculele dificile pentru $1<\nu<2$.
- Estimatorul *"Fair"* are pretitundeni derivate până la ordinul trei inclusiv şi produce soluţii unice. Eficienţa asimptotică de 95% pe distribuţii normale standard este atinsă pentru c=1.345.

- Estimatorul bazat pe funcţia *Hubbe*r, o parabolă în apropiere de origine si liniară dincolo de un prag *k* pentru valoarea absolută a variabilei *x,* atinge eficienţa asimptotică de 95% pentru *k*=1,345.

- Estimatorul este atât de eficient încât este recomandat aproape în toate situaţiile. Totuşi, uneori apar dificultăţi probabil din cauza lipsei de stabilitate în valoare a gradientului legată de discontinuitatea derivatei a doua în punctele $x=\pm k$

$$\frac{d^2\rho(x)}{dx^2} = \begin{array}{ll} 1 & pentru \quad |x| \quad k \\ 0 & pentru \quad |x| > k \end{array} \qquad (6.5)$$

  Eficienţa asimptotică de 95% pe distibuţii normale standard este atinsă pentru *k*=1,2107.
- Funcţia *Cauchy* – cunoscută şi sub numele de funcţia lui *Lorentz* – nu garantează soluţia unică în problema estimării. Cu derivata scăzătoare, funcţia are tendinţa de a produce soluţii eronate într-o manieră inobservabilă. Constanta regulatoare *c* se recomandă a avea valoarea 2.3849.
- Celelalte funcţii au aceleaşi probleme ca şi funcţia *Cauchy*. Graficele indică o descreştere a influenţei erorilor mari doar liniară cu mărimea lor. Funcţile *German-McClure* şi *Welsh* sunt o încercare de a reduce si mai mult efectul erorilor grosiere, funcţia *Tukey dublu ponderată* eliminand chiar punctele străine (outliers). Eficienţa asimptotică a funcţiei *Tukey dublu ponderată,* la 95% pe distribuţia standard normală, se obţine cu constanta de acordare *c*=4.6851 iar pentru funcţia *Welsh* cu *c*=2.9846.

Pare a fi dificil a alege o funcţie ρ pentru uz generalizat fără a introduce o doză de arbitrar. Urmând ideea lui *Rey*, pentru problemele de localizare şi de regresie cea mai bună alegere este funcţia L_p, care în pofida non-robusteţei teoretice, practic se dovedeşte cvasi-robustă. Sunt însă dificultăţi de calcul. Următoarea este funcţia *Fair* care produce proceduri de calcul elegant convergente. Urmează apoi funcţia lui *Hubber*, în forma originară

sau în forma modificată. Toate aceste funcţii nu elimină complet influenţa erorilor grosiere importante.

Tabelul 6.1.

Tip	**ρ(x)**	**ψ(x)**	**w(x)**
L_2	$\frac{x^2}{2}$	x	1
L_1	$\lvert x\rvert$	$\mathrm{Sgn}(x)$	$\frac{1}{\lvert x\rvert}$
$L_2 - L_1$	$2\left(\sqrt{1+\frac{x^2}{2}}-1\right)$	$\frac{x}{\sqrt{1+\frac{x^2}{2}}}$	$\frac{1}{\sqrt{1+\frac{x^2}{2}}}$
L_p	$\frac{\lvert x\rvert^{v}}{v}$	$\mathrm{sgn}(x)\lvert x\rvert^{v-1}$	$\lvert x\rvert^{v-2}$
Fair	$c^2\left[\frac{\lvert x\rvert}{c}-\log\left(1+\frac{\lvert x\rvert}{c}\right)\right]$	$\frac{x}{1+\frac{\lvert x\rvert}{c}}$	$\frac{1}{1+\frac{\lvert x\rvert}{c}}$
Hubber $\lvert x\rvert \le k$ $\lvert x\rvert > k$	$\frac{x^2}{2}$ $k\lvert x\rvert \quad \frac{k^2}{2}$	x $k\,\mathrm{sgn}(x)$	
Cauchy	$\frac{c^2}{2}\log\ 1+\ \frac{x^2}{2}$	$\frac{x}{1+\ \frac{x}{c}^2}$	$\frac{1}{1+\ \frac{x}{c}^2}$
Geman-McClure	$\frac{\frac{x^2}{2}}{1+x^2}$	$\frac{x}{(1+x^2)^2}$	$\frac{1}{(1+x^2)^2}$
Welch	$\frac{c^2}{2}\ 1\ \exp\ \ \frac{x}{c}^2$	$x\exp\ \ \frac{x}{c}^2$	$\exp\ \ \frac{x}{c}^2$

Tukey	$\frac{c^2}{6}\ 1\quad 1\quad \frac{x}{c}^{\ 2\ 3}$ $\frac{c^2}{6}$	$x\ 1\quad \frac{x}{c}^{\ 2\ 2}$ 0	$1\quad \frac{x}{c}^{\ 2\ 2}$ 0

Ultimele patru funcţii prezentate nu garantează unicitatea soluţiei problemei de estimare dar reduce considerabil sau chia elimină complet influenţa erorilor grosiere de amploare. Cum propune *Hubber,* în aceste cazuri se pote începe cu o funcţie ρ convexă, se aduce calculul la convergenţă şi apoi se execută câteva iteraţii cu una din funcţiile non-convexe pentru a elimina efectele erorilor foarte mari.

3. MODUL DE LUCRU

- Dacă nu este deja creat, se creează un director/folder de lucru.
- Se transferă fişierele specifice lucrării prezente, *date luc.mat*, *estimatorm.m*, *filuc4.m*, de pe dischetă sau din directorul corespunzător din reţea, în directorul de lucru.
- Se activează platforma **MATLAB** şi se introduc succesiv secvenţele de program menţionate mai sus.
- Se reprezintă grafic funcţiile din tabelul cuprins în secţiunea **Breviar teoretic** şi se observă variaţia ponderilor, convexitatea funcţiilor de influenţă şi alte aspecte legate de eficienţa şi robusteţea estimatorilor M corespunzători. Un exemplu pentru funcţia L_2 este următorul:

```
function [rho,psi,w]=L2(x)
rho=x.^2/2;
psi=x;
w=ones(size(x));
subplot(1,3,1)
plot(x,rho)
axis([-3 3 0 4.5])
title('rho(x)')
```

```
subplot(1,3,2)
plot(x,psi)
axis([-3 3 -3 3])
title('psi(x)')
subplot(1,3,3)
plot(x,w)
axis([-3 3 0 3])
title('w(x)')
```

Se iniţializează un vector de valori ale variabilei *t*

```
t=-3:0.02:3;
```

Se apelează (repetat)

```
L2(t);
```

- Se apelează *help estimatorm* pentru a observa condiţiile în care *script*-ul *estimatorm* se poate apela şi care sunt funcţiile lui.
- După selectarea unei funcţii se apelează *script*-ul. Se observă graficele generate şi se apreciază calităţile estimatorului după criteriile menţionate în **Breviarul teoretic**. Figura ultimă din cele trei (Stabilitate) ilustrează diferenţele modelelor obţinute în situaţiile când: a) datele au erori normale; b) datele sunt afectate de erori grave sistematice.
- Se cere implementarea software şi a celorlalte tipuri de funcţii din cele menţionate în **Breviarul** teoretic, urmată de analiza şi observaţiile care se impun.
- Vor fi propuse şi funcţii la libera alegere a studenţilor, tot în condiţiile menţionate în Breviarul teoretic: convexitate, mărginire a influenţei erorilor mari, garantarea soluţiei unice. Se cere, de asemenea, formularea unor concluzii consecutive analizei funcţiilor propuse.

LUCRAREA 7

INTERPOLAREA ŞI APROXIMAREA DATELOR

1. OBIECTIVELE LUCRĂRII

- Studiul metodelor de aproximare a datelor prin regresie liniară şi polinomială;
- Studiul metodelor de interpolare liniară prin metoda transformatei Fourier.

2. BREVIAR TEORETIC

Se propune următoarea strategie de aproximare: mai întâi, aproximarea unui set de date printr-o linie dreaptă (*regresie liniară*), apoi prin aproximarea printr-un polinom (*regresie polinomială*).

Pentru ca aproximarea să fie considerată foarte bună, suma pătratelor distanţelor de la fiecare punct la curba aproximată (linie sau polinom) trebuie să fie minimă. Cu această condiţie îndeplinită, este posibil ca nici un punct al setului de date să nu se găsească pe curba aproximantă, ceea ce reprezintă diferenţa faţă de interpolare, la care toate punctele sunt situate pe curbă.

2.1. Regresie liniară

Regresia liniară aproximează setul de date printr-o dependenţă liniară care minimizează suma pătratelor dintre dreapta de aproximare şi punctele date.

Măsura calităţii unei aproximări liniare, dată de suma pătratelor distanţelor de la fiecare punct la estimaţia liniară, este exprimată prin

$$sum\ p = sum\ ((y-y_1).\hat{}2) \qquad (7.1)$$

Determinarea parametrilor *m* şi *n* ai dreptei de aproximare $y = mx+n$ se face folosind funcţia **polyfit**.

Exemplul 1.
Să se aproximeze în sensul CMMP cu o regresie liniară setul de date:

x=[0,1,2,3,4,5]
y=[0,20,60,68,77,100].

Secvenţa MATLAB care realizează această aproximare este prezentată în continuare.

```
x=[0,1,2,3,4,5];
y=[0,20,60,68,77,100];
coef=polyfit(x,y,1);
m=coef(1);
n=coef(2);
y1=m*x+n;
sump=sum((y-y1).^2);
axis([-1,6,-20,120]);
plot(x,y1,x,y,'0');
grid on
```

Interpretaţi reprezentarea grafică obţinută.

2.2. Regresie polinomială

Regresia polinomială realizează aproximarea setului de date printr-un polinom de forma

$$p(x)=\sum_{i=0}^{N} a_i x^{N-i} = a_0 x^N + a_1 x^{N-1} + \ldots + a_{N-1}x + a_N \qquad (7.2)$$

Coeficienţii $a_0,a_1,...a_N$ ai polinomului de regresie sunt calculaţi cu ajutorul metodei CMMP, ceea ce presupune minimizarea funcţiei obiectiv

$$f_{ob}(A) = \sum_{i=1}^{m}[y_i - (a_0 x_i^N + a_1 x_i^{N-1} + ... + a_N)]^2 \qquad (7.3)$$

Algoritmul regresiei polinomiale cuprinde următoarele etape:

1. Calculul sumelor în x_i şi y_i;

$$\begin{aligned} &Sx_1 = m; \\ &Sx_j = \sum_{i=1}^{m} x_i^{j-1},\ j = 2,...2n_{\max}+1; \\ &Sxy_1 = \sum_{i=1}^{m} y_i; \\ &Sxy_j = \sum_{i=1}^{m} y_i x_i^{j-1},\ j = 2,...,n_{\max}+1. \end{aligned} \qquad (7.4)$$

2. Calculul polinomului de regresie de grad n.

3. Generarea matricii sistemului de ecuaţii, a termenului liber şi rezolvarea sistemului de ecuaţii liniare

$$\begin{aligned} &c_{ij} = Sx_{i+j-1},\ j = 1,...n+1, i = 1,...,n+1; \\ &b_j = Sxy_j,\ j = 1,...,n+1; \\ &CA = B \\ &w_{ni} = a_j,\ j = 1,...,n+1 \end{aligned} \qquad (7.5)$$

4. Calculul dispersiei şi a abaterii standard

$$\begin{aligned} &s^2 = \sum_{i=1}^{m}[y_i - (a_0 x_i^N + ... + a_N)]^2; \\ &\sigma = \sqrt{s^2}; \\ &disp_n = s^2 \end{aligned} \qquad (7.6)$$

5. Testarea gradului polinomului de regresie în vederea validării lui

$n<n_{max} \Rightarrow n=n+1$ şi salt la 3;

$n>n_{max} \Rightarrow$ salt la 6.

6. Determinarea gradului optim *j* al polinomului de regresie

$$\min_{j} (grad_j,\ j=1,\ldots,n_{max})$$

Dacă setul de date are N elemente, toate datele se află pe curba de aproximare. Pentru un polinom având gradul mai mic decât numărul de date, aproximarea este cu atât mai bună cu cât gradul polinomului este mai apropiat de numărul de date. Utilizarea unui polinom de aproximare de grad mai mare decăt setul de date poate conduce la erori de aproximare considerabile.

Determinarea celei mai bune aproximări a unui set de date (*x,y)* cu un polinom de ordinul *n* se face folosind funcţia **polyfit**, cu sintaxa

```
p=polyfit(x,y,n)
```

care returnează coeficienţii a_i ai polinomului *p(x)* caracterizat de proprietatea că prezintă, în punctele precizate de vectorul *x*, valorile date de vectorul *y* (în sensul CMMP).

Exemplul 2

Fie polinomul $p(x)=x^3-6x^2+11x-6$, peste care este suprapus un zgomot cu distribuţie normală. Aproximaţi în sensul CMMP datele rezultate cu un polinom de grad 3. Reprezentaţi grafic datele cu zgomot şi polinomul aproximant.

Secvenţa de program MATLAB este

```
p=[1,-6,11,-6];
x=0:.25:4;
y=polyval(p,x)+rand n(size(x));
c=polyfit(x,y,3);
poli3=polyval(c,x);
plot(x,poli3,x,y,'0');
grid on
```

Obs. Funcţia **polyval(p,s)** evaluează polinomul definit de vectorul *p*, al coeficienţilor polinomiali, în punctul *s*.

2.3.Interpolarea funcţiilor de o singură variabilă

2.3.1. Interpolarea prin metoda transformatei Fourier

Funcţia **interpft** interpolează datele cu o singură variabilă utilizând metoda FFT şi se apelează cu sintaxa

y=interpft(x,n)

care returnează un vector *y* de lungime *n* obţinut din vectorul *x*. Numărul *n* trebuie să fie mai mare decât numărul de elemente al vectorului *x*, iar rezultatul are o periodicitate circulară dată de utilizarea transformatei Fourier.

Exemplul 3

Fie datele obţinute dintr-un sinus eşantionat cu 8 paşi pe perioadă

$$y=\sin\frac{2\pi k}{8} \qquad (7.7)$$

Interpolaţi datele cu un pas dublu şi verificaţi valorile obţinute cu secvenţa MATLAB.

```
k=0:7;
x=sin(2*pi*k/8);
yi=interpft(x,16);
k1=0:15;
yr=sin(2*pi*k1/16);
d=max(yi-yr);
```

Rezultatul obţinut este

$d=5.5511e^{-0,16}$

Diferenţa maximă între valorile interpolate şi cele reale este egală cu ordinul de mărime al celui mai mic număr reprezentabil în calculator (eroarea de trunchiere).

2.3.2.Interpolarea liniară

Dacă se presupune că funcţia dintre două puncte de coordonate (x_1, y_1), respectiv (x_2, y_2), poate fi estimată printr-o linie dreaptă, atunci

valoarea funcţiei în orice punct *x* dintre cele două valori se deduce cu expresia

$$f(x)= y_1 + \frac{x - x_1}{x_2 - x_1}\left(y_2 - y_1\right) \qquad (7.8)$$

Interpolarea liniară a funcţiilor de o singură variabilă se face cu funcţia *tabel* care se apelează cu sintaxa

y=tabel(nume_tabel,x)

Primul argument al funcţiei este numele tabelului care conţine datele ce se referă la coordonatele (x_i,y_i). Dacă acesta este un fişier pe disc, atunci *nume_fişier* trebuie să fie mai întâi încărcat cu funcţia *load*.

Al doilea argument se referă la valorile lui *x* pentru care se doreşte determinarea valorilor interpolate *y*.

Datele din prima coloană a tabelului (valorile lui *x*) trebuie să fie în ordine crescătoare, iar valorile *x* trebuie să se găsească între prima şi ultima valoare a primei coloane; în caz contrar, se afişează un mesaj de eroare.

Dacă tabelul din care se citesc datele conţine mai mult de două coloane, funcţia *tabel* returnează un vector linie cu *N-1* elemente, unde *N* este numărul de coloane ale tabelului. Fiecare valoare returnată este interpolată din coloana corespunzătoare a datelor.

Exemplul 4.
Estimaţi valorile temperaturii la momentele de timp 2.5 sec şi 4.9 sec, cu datele următoare:

0.0℃ ..... 0.0 s
20.0℃ ..... 1.0 s
60.0℃ ..... 2.0 s
68.0℃ ..... 3.0 s
77.0℃ ..... 4.0 s
110.0℃ .....5.0 s

Se introduc datele într-o matrice care are în prima coloană valorile timpului şi în a doua coloană temperaturile corespunzătoare:

temp1(:,1)=[0.0,1.0,2.0,3.0,4.0,5.0]';
temp1(:,2)=[0.0,20.0,60.0,68.0,77.0,110.0]';

Cu instrucţiunea

y=tabel(temp1,[2.5 4.9])

se obţin rezultatele

y=[64.0 107.0]

3. MODUL DE LUCRU. PROBLEME PROPUSE

- Dacă nu este deja creat, se creează un director/folder de lucru.
- Se activează platforma **MATLAB** şi se introduc secvenţele de program prezentate în exemplele din cuprinsul **Breviarului teoretic,** urmărindu-se îndeplinirea cerinţelor enunţate la fiecare dintre ele.
- Se consideră un traductor de debit cu diafragmă. Setul de date experimentale obţinute pentru trasarea caracteristicii statice, şi anume mărimea de intrare - debitul volumic - şi mărimea de ieşire - curentul generat de traductor - sunt următoarele:

10,14 m^3/h……10 mA
8,64 m^3/h ……..8 mA
7,20 m^3/h……6 mA
4,5 m^3/h…….3,5 mA
2 m^3/h……..2,5 mA
1,5 m^3/h……..2,2 mA
0,1 m^3/h……..2 mA

Se cere aproximarea setului de date printr-un polinom de forma prezentată în relaţia (7.2), pe baza algoritmului de la paragraful 2.2.

LUCRAREA 8

IDENTIFICAREA SISTEMELOR FOLOSIND METODA CELOR MAI MICI PĂTRATE. ALGORITMI DE CALCUL

1. OBIECTIVELE LUCRĂRII

Estimarea parametrilor unui model matematic folosind metoda celor mai mici pătrate (**MCMMP**).

2. BREVIAR TEORETIC

În cadrul metodei celor mai mici pătrate (MCMMP), sistemul se consideră descris de următoarea ecuaţie cu diferenţe

$$A(q^{-1})y(t)=B(q^{-1})u(t)+e(t) \tag{8.1}$$

unde :

$u(t)$=mărimea de intrare ;
$y(t)$=mărimea de ieşire ;
$e(t)$=zgomot alb de medie zero şi dispersie λ^2 ;
q^{-1}=operator de întârziere

Modelul se consideră descris de o ecuaţie cu diferenţe, care are aceeaşi structură cu ecuaţia (8.1)

$$A_T(q^{-1})y(t)=B_T(q^{-1})u(t)+e(t) \tag{8.2}$$

unde :

$e(t)$=reziduul modelului ;
$A_T(q^{-1})=1+a_{1T}q^{-1}+ a_{2T}q^{-2}+\ldots\ldots+ a_{naT}q^{-na}$;
$A_T(q^{-1})=1+b_{1T}q^{-1}+ b_{2T}q^{-2}+\ldots\ldots+ b_{nbT}q^{-nb}$.

Se fac următoarele notaţii:

$$\theta=[a_{1T}\ldots\ldots\ a_{naT,}\ b_{1T}\ldots\ldots\ldots\ b_{nbT}]^T$$
$$\varphi(t)=[-y(t-1)\ldots\ldots\ldots\ -y(t-na)\ u(t-1)\ldots\ldots\ldots\ u(t-nb)]^T\ ; \qquad (8.3)$$

cu aceste notaţii mărimea de ieşire dată de model fiind

$$y_m(t)=\varphi^T(t)\theta+\varepsilon(t). \qquad (8.4)$$

Având disponibilă structura modelului (*na*,*nb*), se impune condiţia ca media pătratică a erorii de predicţie să fie minimă. Estimaţia celor mai mici pătrate a lui θ, bazată pe *n* date de definiţie, este

$$\hat{\theta} = \arg\min_{\bar{\theta}} V\left(\bar{\theta}\right), \qquad (8.5)$$

unde $V\left(\bar{\theta}\right)= \sum_{i=1}^{n}\left[y(t)-\varphi^{-T}(t)\bar{\theta}\right]^2$ (8.6)

Din condiţia de mai sus rezultă

$$\hat{\theta} = \left(\sum_{i=1}^{N}\bar{\varphi}(t)\bar{\varphi}^T(t)\right)^{-1}\sum_{i=1}^{N}\bar{\varphi}(t)y(t). \qquad (8.7)$$

Cu notaţiile :

$\theta=[a_{1T}\ldots\ldots\ a_{naT,}\ b_{1T}\ldots\ldots\ldots\ b_{nbT}]^T$: parametrii modelului
$\varphi(t)=[-y(t-1)\ldots\ldots\ldots\ -y(t-na)\ u(t-1)\ldots\ldots\ldots\ u(t-nb)]^T$: vectorul care conţine istoria procesului (intrările şi ieşirile anterioare),

ecuaţia (8.1) devine

$$y(t)=\varphi^T(t)\theta+\varepsilon(t). \qquad (8.8)$$

Modelul va fi descris printr-o ecuaţie de aceeaşi formă

$$y_m(t)=\varphi^T(t)\theta+\varepsilon(t). \qquad (8.9)$$

Estimarea parametrilor modelului (θ) presupune în primul rând stabilirea gradelor *na* şi *nb* şi apoi determinarea vectorului θ pe baza datelor experimetale de intrare-ieşire.

De fapt, esenţa metodei constă în a presupune faptul că modelul este determinist

$$y_m(t)=\varphi^T(t)\theta\,, \tag{8.10}$$

situaţie în care θ se calculează impunându-se următoarea condiţie

$$\theta=\arg_\theta\min\frac{1}{N}\sum_{t=1}^{N}(y(t)-y_m(t))^2=\arg_\theta\min\frac{1}{N}\sum_{t=1}^{N}(y(t)-\varphi^T(t)\theta)^2 \tag{8.11}$$

Expresia explicită a lui $\widehat{\theta}$ se obţine din condiţia de anulare a gradientului funcţiei criteriu

$$V_{\bar{\theta}}\left(\hat{\theta}\right)=0\,,\ \frac{\partial V}{\partial\bar{\theta}}\Big|_{\bar{\theta}=\hat{\theta}}=-2\frac{1}{N}\sum_{t=1}^{N}\bar{\varphi}(t)\left(y(t)-\bar{\varphi}^T(t)\hat{\theta}\right)^2=0$$

$$\sum_{t=1}^{N}\bar{\varphi}(t)y(t)=\sum_{t=1}^{N}\bar{\varphi}(t)\bar{\varphi}^T(t)\hat{\theta}\ \Rightarrow\hat{\theta}=\left[\sum_{t=1}^{N}\bar{\varphi}(t)\bar{\varphi}^T(t)\right]^{-1}\sum_{t=1}^{N}\bar{\varphi}(t)y(t)\,. \tag{8.12}$$

Dacă se notează

$$Y=\left[y(1)\ldots y(N)\right]^T,\ \Phi^T=\left[\bar{\varphi}(1)\ldots\bar{\varphi}(N)\right], \tag{8.13}$$

atunci se obţine

$$\Phi^T\Phi=\sum_{t=1}^{N}\bar{\varphi}(t)\bar{\varphi}^T(t) \tag{8.14}$$

$$\Phi^TY=\left[\bar{\varphi}(1)\ldots\bar{\varphi}(N)\right]\begin{bmatrix}y(1)\\ \vdots\\ y(N)\end{bmatrix}=\sum_{t=1}^{N}\bar{\varphi}(t)y(t) \tag{8.15}$$

$$Y=\Phi\theta+\varepsilon\,. \tag{8.16}$$

Cu aceste notaţii estimatorul se poate scrie sub forma

$$\hat{\theta}=\left[\Phi^T\Phi\right]^{-1}\Phi^TY \tag{8.17}$$

Estimatorul dat prin relaţia de mai sus reprezintă estimatorul celor mai mici pătrate care s-a obţinut pe baza datelor de intrare $u(1),\ldots,u(N)$ şi a celor de ieşire $y(1),\ldots,y(N)$.

Proprietăţile estimatorului celor mai mici pătrate

Estimatorul celor mai mici pătrate este un predictor al mărimii de ieşire la momentul *t* dedus pe baza datelor de ieşire până la momentul *(t-1)* (*predictor de pas*).

$$y_m(t)=\varphi^T(t)\theta,$$
$$\varphi(t)=[-y(t-1)\ldots\ldots -y(t-na)\ u(t-1)\ldots\ldots\ u(t-nb)]^T \qquad (8.18)$$

deci $y_m(t)=y(t/t-1),$

de unde rezultă că *y(t)* este determinat pe baza datelor de intrare/ieşire până la momentul (*t-1*).

O altă proprietate a estimatorului celor mai mici pătrate se referă la faptul că eroarea de predicţie $\frac{1}{N}\sum_{t=1}^{N} q^2(t)$ este minimă.

Această afirmaţie se justifică având în vedere relaţiile următoare

$$\frac{1}{N}\sum_{t=1}^{N} q^2(t)=\frac{1}{N}\sum_{t=1}^{N}(y(t)-y_m(t))^2 \qquad (8.19)$$

$$\theta=\arg_\theta \min\frac{1}{N}\sum_{t=1}^{N}(y(t)-\varphi^T(t)\theta)^2 \qquad (8.20)$$

3. MODUL DE LUCRU

- Dacă nu este creat, se creează directorul de lucru ;
- Se activează platforma de lucru **MATLAB** ;
- Se consideră modelul descris de următoarea ecuaţie cu diferenţe

$$y(t)-1.5y(t-1)+0.7y(t-2)=u(t-1)+0.5u(t-2)+e(t)$$

- Se va utiliza următoarea secvenţă de program :

```
a=[1 -1.5 0.7];
b=[0 1 0.5];
c=[1];
d=[1];
f=[1];
n1=200;
k=10;
ths=poly2th(a,b,c,d,f);
u=sign(randn(n1,1));
e=randn(n1,1)/k;
 ys=idsim([u e],ths);
z=[ys u e]
 idplot([ys u]);
```

unde *k* este factorul de ponderare al erorii, iar *=[ys u e]* reprezintă o matrice cu trei coloane ce conţine ieşirea (*ys*), intrarea (*u*) şi perturbaţia (*e*).

- După execuţia secvenţei de mai sus, intrarea şi ieşirea sistemului vor arăta ca în figura 8.1., unde în graficul din partea superioară a fost reprezentată secvenţa datelor de ieşire, iar în graficul din partea inferioară secvenţa datelor de intrare.

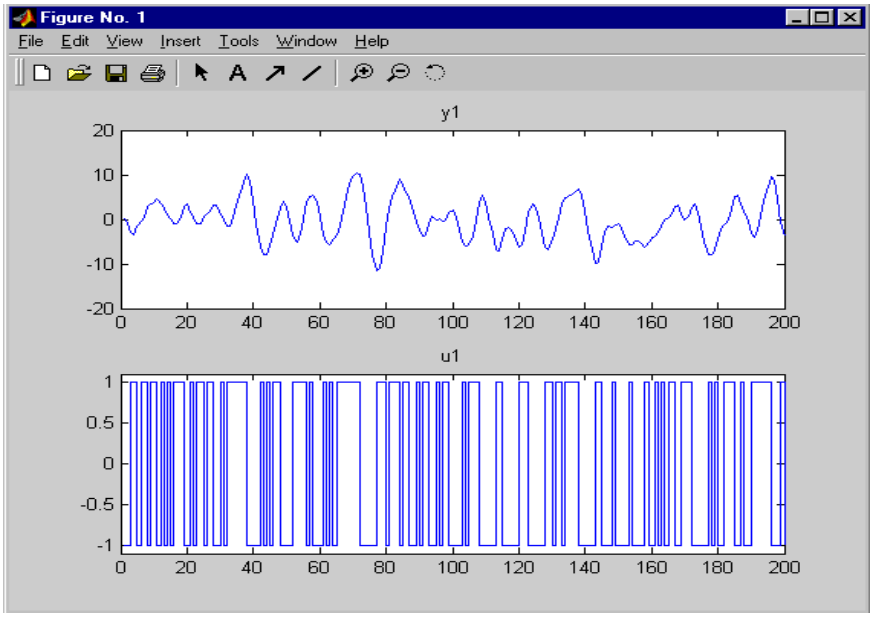


Figura 8.1. Reprezentările grafice ale intrării şi ieşirii sistemului descris de modelul din relaţia 8.27.

- Se elaborează programul care implementează etapa de estimare şi se verifică obţinerea modelului (considerând *na=2* şi *nb=2*) de forma

y(t)-1.496y(t-1)+0.703y(t-2)=0.997u(t-1)+0.523u(t-2)+e(t)

şi a unei erori medii de 0.0183.

- Se modifică structura propusă şi se repetă operaţia de estimare de parametri şi de determinare a erorii medii pătratice.

- Se verifică faptul că prin creşterea gradelor polinoamelor modelului, eroarea medie pătratică nu scade considerabil.

LUCRAREA 9

IDENTIFICAREA EXPERIMENTALĂ A SISTEMELOR DE ORDINUL I ŞI ORDINUL II

1. OBIECTIVELE LUCRĂRII

Identificarea unor sisteme de diverse ordine de mărime prin urmărirea evoluţiei în timp a ieşirii în condiţiile aplicării la intrarea sistemelor a unor semnale tip: treaptă, rampă, sinusoidă;

2. BREVIAR TEORETIC

2.1.Sisteme de ordinul întâi

Modelul unui sistem de tip continuu, liniar, invariant, monovariabil şi cu parametri concentraţi este

$$y^{(n)} + a_{n-1}y^{(n-1)} + \ldots + a_1\dot{y} + a_0 y = b_r u^{(r)} + b_{(r-1)}u^{(r-1)} + \ldots + b_1\dot{u} + b_0 u, \tag{9.1}$$

unde: u= mărime de intrare; y= mărime de ieşire.
În regim staţionar se obţine modelul

$$a_0 y = b_0 u, \tag{9.2}$$

care conduce la obţinerea unui răspuns de forma

$$y = \left(b_0 / a_0\right)u \tag{9.3}$$

În ceea ce priveşte ordinul maxim de derivare al intrării şi al ieşirii, se deosebesc trei cazuri:

-dacă r<n, sistemul este strict propriu;
-dacă r=n, sistemul este simplu propriu;
-dacă r>n, sistemul este impropriu.

Sistemele de ordinul întâi sunt descrise de ecuaţia

$$a_1 \dot{y} + a_0 y = b_0 u, \tag{9.4}$$

Rezolvând această ecuaţie în domeniul timpului se obţine următorul răspuns al sistemului

$$y(t) = b_0 u\left(1 - e^{-(a_0/a_1)t}\right) \tag{9.5}$$

Notând cu $1/T = a_0/a_1$ şi cu $y_0 = (b_0/a_0)u$, se obţine următoarea expresie a ieşirii:

$$y(t) = y_0\left(1 - e^{-t/T}\right) \tag{9.6}$$

unde:

T = constanta de timp a sistemului;
y_0 = răspunsul sistemului în regim staţionar;

Durata regimului tranzitoriu este $T_{tr} = 3T$ şi exprimă timpul în care răspunsul sistemului ajunge la 95% din valoarea răspunsului în regim staţionar.

Pentru sistemele de ordinul întâi fără anticipare (ordinul maxim de derivare al intrarii este zero), se deosebesc urmatoarele cazuri:

a) $a_0 \neq 0, b_0 \neq 0$, modelul staţionar este

$$y_0 = (b_0/a_0)u \tag{9.7}$$

şi exprimă sistemul de tip *proporţional*;

b) $a_0 = 0, b_0 \neq 0$, sistemul este de tip *integrator* cu următorul model

$$a_1 \dot{y} = b_0 u, \tag{9.8}$$

având răspunsul

$$y(t) = (b_0/a_0)\int_0^t u\,dt \tag{9.9}$$

În cazul în care ordinul de derivare al intrării este 1 şi este mai mare decât ordinul de derivare al ieşirii (sistemul este impropriu), se mai poate deosebi un al treilea caz care este clar idealizat deoarece toate sistemele fizice sunt sisteme proprii.

c) $a_0 \neq 0, b_0 = 0$, sistemul este de tip *derivator*, cu modelul

$$a_0 y = b_1 \dot{u}, \tag{9.10}$$

care are răspunsul: $y = \left(b_1 / a_0\right)\left(du / dt\right)$ (9.11)

Funcţia de transfer a unui sistem de ordinul întâi fără anticipare este

$$H(s) = b_0 / \left(a_1 s + a_0\right) \tag{9.12}$$

Graficele răspunsului în timp ale sistemelor de ordinul întâi la diferite tipuri de intrări: treaptă, rampă şi sinusoidă sunt reprezentate în figura 9.1.

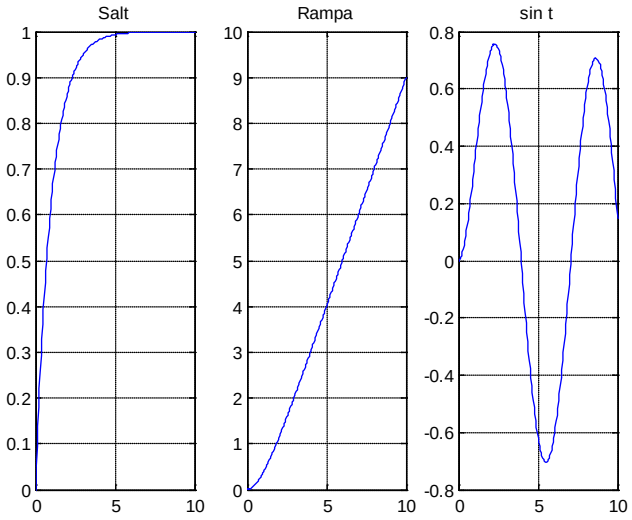


Fig.9.1. Răspunsuri în timp ale sistemului de ordinul întâi la diferite tipuri de mărimi de intrare: treaptă, rampă şi sinusoidă

Practic, un sistem de ordinul întâi se poate obţine cu un circuit RC, reprezentat în figura 9.2.

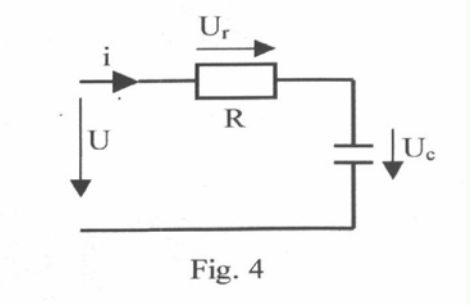


Fig.9.2.. Circuit RC

Modelul matematic este prezentat în setul (9.13)

$$\begin{cases} U = U_r + U_c \\ U_r = Ri \\ U_c = (1/C)\int idt \Rightarrow i = C\dot{U}_c \\ U_r = RC\dot{U}_c \end{cases} \tag{9.13}$$

Ecuaţia care descrie sistemul de ordinul întâi este

$$RC\dot{U}_c + U_c = U \tag{9.14}$$

unde: U_c =căderea de tensiune pe condensator;

U_r =căderea de tensiune pe rezistenţă;

U=tensiunea de alimentare a circuitului RC;

I=intensitatea curentului prin circuit;

R=rezistenţă, C=condensator.

Intrarea sistemului a fost considerată tensiunea de alimentare a circuitului RC, iar ieşirea sistemului a fost considerată tensiunea măsurată la bornele condensatorului C (*Uc*). Schema bloc a sistemului de ordinul întâi este prezentată în figura 9.3.

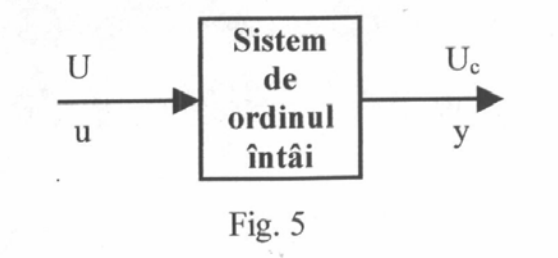


Fig. 9.3.. Schema bloc a sistemului de ordinul întâi

2.2 Sisteme de ordinul doi

Sistemele continue de ordinul doi fără anticipare sunt descrise de ecuaţia

$$a_2 \ddot{y} + a_1 \dot{y} + a_0 y = b_0 u, \tag{9.15}$$

Rezolvând ecuaţia în domeniul timpului se obţine răspunsul sistemului

$$y_l(t) = c_1 e^{-A/t} + c_2 e^{-B/t} + b_0 u, \tag{9.16}$$

cu c_1 si c_2 două constante determinate din condiţiile iniţiale, iar *A, B* soluţii ale ecuaţiei

$$a_2 r^2 + a_1 r + a_0 = 0 \tag{9.17}$$

Componenta $y_l(t) = c_1 e^{-A/t} + c_2 e^{-B/t}$ se numeşte *componentă liberă*, dependentă numai de structura sistemului, neinfluenţată de forma de variaţie a intrării, iar componenta $y_f(t) = (b_0 / a_0) u$, se numeşte *componentă forţată,* numită astfel deoarece este rezultatul "forţării" in forma răspunsului a unei componente de tipul intrării.

Funcţia de transfer a unui sistem de ordinul doi fără anticipare este

$$H(s) = b_0 / (a_2 s^2 + a_1 s + a_0) \tag{9.18}$$

Graficele răspunsurilor în timp ale sistemelor de ordinul doi la diferite intrări tip: treaptă, rampă, sinusoidă sunt prezentate în figura 9.4.

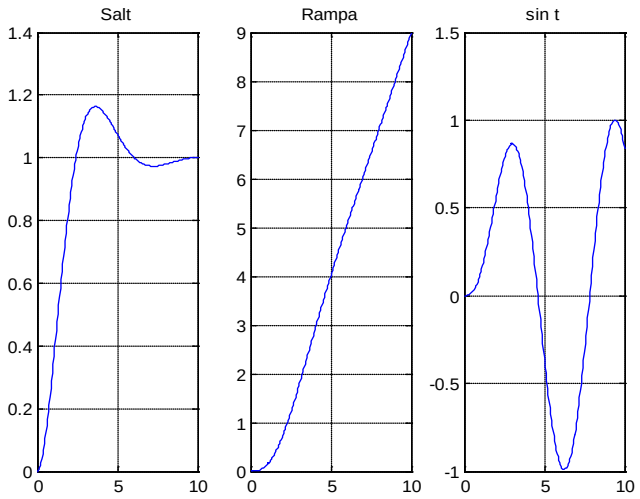


Fig.9.4. Graficele răspunsurilor în timp ale sistemelor de ordinul doi la diferite tipuri de mărimi de intrare: treaptă, rampă şi sinusoidă

Practic, un sistem de ordinul doi se poate obţine cu un circuit RLC de tipul celui din figura 9.5.

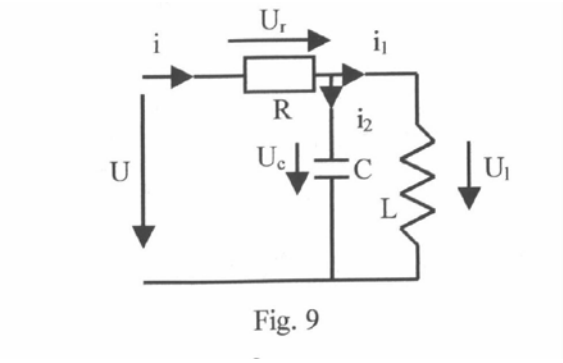


Fig.9.5. Circuit RLC

Modelul matematic este

$$\begin{cases} U_r = Ri \\ U_c = (1/C)\int i2dt \Rightarrow i2 = C\dot{U}_c \\ Ul = i1 + i2 \Rightarrow LC\ddot{U}\,l + (L/R)\dot{U}\,l = (L/R)U \end{cases} \tag{9.19}$$

Se obţine astfel un model de forma

$$LC\ddot{U}\,l + (L/R)\dot{U}\,l + Ul = (L/R)U \tag{9.20}$$

unde:

U=tensiunea de alimentare a circuitului RLC;

U_r =căderea de tensiune pe rezistenţă;

U_c =căderea de tensiune pe condensator;

U_l =căderea de tensiune pe bobine;

i=intensitatea curentului prin rezistenţă;

I_1 =intensitatea curentului prin bobine;

I_2 =intensitatea curentului prin condensator;

R=rezistenţă;

C=condensator;

L=bobină.

Intrarea sistemului a fost considerată tensiunea de alimentare a circuitului RLC, U, în timp ce ieşirea a fost considerată căderea de tensiune pe bobina L, U_1.

Schema bloc a sistemului este reprezentată în figura 9.6.

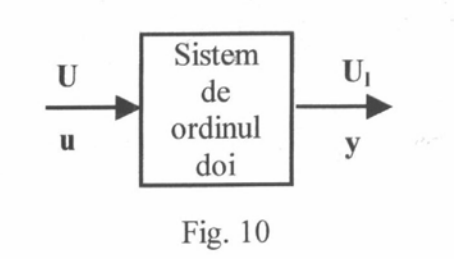


Fig. 10

Fig.9.6. Schema bloc a sistemului de ordinul doi.

3. MODUL DE LUCRU

Schema bloc a montajului utilizat pentru identificarea sistemelor de diverse ordine de mărime este reprezentată în figura 9.7.

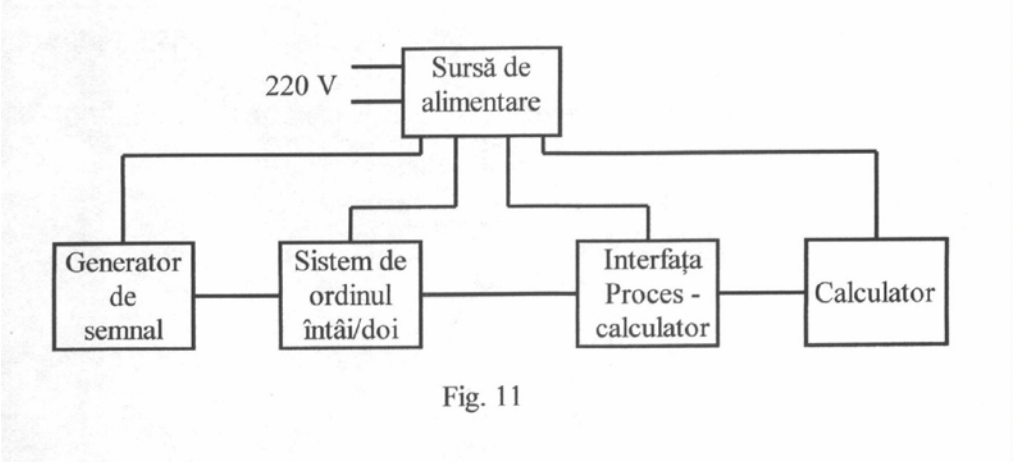


Fig. 11

Fig. 9.7. Schema bloc a montajului utilizat pentru identificarea experimentală a sistemelor.

1. Conform schemei din figura 9.7, se interconectează blocurile componente, cu următoarele observaţii:
   - se vor respecta regulile de conectare a interfeţei la alimentarea cu 5V (plusul, respectiv masa, de la sursa de alimentare, la firul de +5V, respectiv de masă, la interfaţă);
   - se va conecta cablul paralel de la interfaţă la portul paralel al PC-ului;
   - se va conecta la intrarea interfeţei ieşirea de la cele două sisteme; acestea vor avea la intrare semnalele de tip treaptă sau sinusoidă date de versatester.
2. Se lansează în execuţie programul aferent (IS.EXE, varianta de DOS, sau IDENT. EXE, varianta de Windows);
3. De la generatorul de semnal versatester tip E0502 se vor emite un semnal sinusoidal, respectiv dreptunghiular, cu frecvenţa de 1 Hz;

4. În urma vizualizarii rezultatelor experimentului, se vor identifica sistemele analizate;
5. La sfârşitul lucrării se vor deconecta elementele componente.

Cu ajutorul generatorului de semnal, studenţii vor aplica la intrarea circuitelor RC si RLC (descrise în partea teoretică), semnale de tip treaptă şi sinusoidă.

Pentru obţinerea unei bune discretizări a semnalului analogic *y*, s-a utilizat un convertor analogic-numeric pe 8 biţi la ieşire. Deoarece portul paralel are doar 4 biţi de intrare a fost necesară utilizarea unui buffer.

Interfaţa "proces" – calculator are la bază un convertor analog-numeric care face posibilă vizualizarea pe monitorul unui calculator a valorilor ieşirii *y* în timp şi a graficelor ieşirii *y* pentru diversele intrări aplicate la intrarea sistemelor.

LUCRAREA 10

METODE DE SUBSPAŢIU ÎN IDENTIFICAREA SISTEMELOR

1. OBIECTIVELE LUCRĂRII

- Identificarea sistemelor prin metode de subspaţiu;
- Obţinerea directă din date experimentale a modelelor cu evidenţierea spaţiului stărilor, adică sub forma ecuaţie de stare – ecuaţie de observare.

2. BREVIAR TEORETIC

Metodele de subspaţiu sunt în câmpul identificării sistemelor de relativ puţin timp (1994) şi aparţin unor cercetători belgieni, Peter Van Overschee şi Bart De Moore. Ele reprezinta o tratare a problemei identificării sistemelor concomitent deterministe şi stochastice, etichetata drept revoluţionară. Foarte repede, această problemă a fost implementată în pachetul de programe **MATLAB,** în subpachetul *Toolbox/Ident*, sub numele ***n4sid***.

Algoritmic, pe baza unor proiecţii ale ieşirilor ″viitoare″ ale sistemului pe intrările ″trecute″ şi ″viitoare″ şi pe ieşirile ″trecute″ (″trecutul″ şi ″viitorul″ sunt delimitate de indicele *i* şi sunt, aşadar, noţiuni convenţionale), problema se tratează în maniera prezentată în continuare.

Pasul 1. Determinarea celor două proiecţii Z_i şi Z_{i+1}

$$Z_i = Y_{i/2i-1}\begin{pmatrix} U_{0/i-1} \\ U_{i/2i-1} \\ Y_{0/i-1} \end{pmatrix} = \left(L_i^1 \cdot L_i^2 \cdot L_i^3\right)\begin{pmatrix} U_{0/i-1} \\ U_{i/2i-1} \\ Y_{0/i-1} \end{pmatrix} \qquad (10.1)$$

$$Z_{i+1} = Y_{\text{i+1/2i-1}} \begin{pmatrix} U_{0/\text{i}} \\ U_{\text{i+1/2i-1}} \\ Y_{0/i} \end{pmatrix} \tag{10.2}$$

cu *U* şi *Y* matrici bloc de intrări şi ieşiri observate experimental între momentele scrise ca indici.

Pasul 2. Determinarea descompunerii prin valori singulare

$$\left(L_i^1 \cdot L_i^3\right)\begin{pmatrix} U_{0/\text{i-1}} \\ Y_{0/\text{i-1}} \end{pmatrix} = \left(U_1 \cdot U_2\right)\begin{pmatrix} \sum_1 & 0 \\ 0 & 0 \end{pmatrix} V^T \tag{10.3}$$

Ordinul sistemului este egal cu numărul de valori singulare nenule

$$\Gamma_i = U_1 \sum_1^{1/2} \text{ şi } \Gamma_{\text{i-1}} = U_1 \sum_1^{1/2} \tag{10.4}$$

Pasul 3. Determinarea soluţiei prin metoda celor mai mici pătrate (cu ρ_1 şi ρ_2 reziduale)

$$\begin{array}{cccccccccc} & & j & & n & mi & & j & & j \\ n & \Gamma_{\text{i-1}}^{\text{p}} Z_{i+1} = & n & K_{11} & K_{12} & n \cdot \Gamma_i^p Z_i & +\text{n} \cdot & n & \rho_1 \\ l & Y_{i/i} & l & K_{21} & K_{22} & mi & U_{\text{i/2i-1}} & \text{l} & l & \rho_2 \end{array} \tag{10.5}$$

Pasul 4. Obţinerea matricilor sistemului succesiv, astfel:

4.1. În prima etapă $A \leftarrow K_{11}$ şi $C \leftarrow K_{12}$

4.2. Se obţin *B* şi *D* care rezultă din *A, C, K_{12}* şi *K_{22}* prin rezolvarea unui sistem de ecuaţii liniare.

4.3. analog, $\begin{pmatrix} Q^S & S^S \\ \left(S^S\right)^T & R^S \end{pmatrix} \leftarrow \frac{1}{j}\begin{pmatrix} \rho_1\rho_1^T & \rho_1\rho_2^T \\ \rho_2\rho_1^T & \rho_2\rho_2^T \end{pmatrix}$

Subsistemul determinist este identificat exact pe măsură ce $j \to \infty$, independent de numărul *i*.

Aproximarea subsistemului stochastic depinde de *i* şi converge pe măsură ce $i \to \infty$.

Tot acest algoritm aparent complicat este implementat în funcţia **MATLAB** cu numele ***n4sid***, sub forma simplă

TH=N4SID(Z)

sau mai complicată

[TH,AO]=N4SID(Z,ORDER,NY,AUXORD,DKX,MAXSIZE,TSMP)

În *TH* se returnează modelul cu evidenţierea stării sistemului, în formatul *THETA*. Funcţia nu furnizează un model al covariaţiilor.

În matricea *Z* sunt plasate datele de ieşire şi de intrare sub forma unor vectori coloană [*y u*]. Dacă sistemul are mai multe intrări *(n)* şi mai multe ieşiri *(p)* atunci *Z=[y1 y2 ... yp u1 u2 ... un].*

Variabila *ORDER* specifică ordinul/ordinele posibil(e) al(e) modelului (dimensiunea vectorului de stare). Introdusă ca vector, de pildă 3:10, ea face ca funcţia să genereze un grafic cu informaţii asupra tuturor ordinelor propuse. Prin *default* ORDER=1:10. Dacă variabila ORDER este introdusă ca 'best', se alege această valoare de *default*. NY este numărul de ieşiri din matricea de date; prin *default* NY=1.

AUXORD este un ordin auxiliar care este folosit la selectarea variabilelor de stare. Prin *default* ea este 1.2*ORDER+3. Dacă AUXORD este introdus ca un vector linie atunci este reţinută acea valoare care minimizează eroarea de predicţie.

Variabila *DKX* este un vector care defineşte structura DKX= [D,K,X]. D=1 indică estimarea unui termen direct de la intrare la ieşire, D=0 indică postularea unei întârzieri de la intrare la ieşire. *K*=1 impune estimarea unei matrici *K*, iar *K*=0 obligă la o matrice *K* nulă. *X*=1 indică o estimare a stării iniţiale a sistemului, *X*=0 indică iniţierea la zero.

Pentru definirea unei structuri de întârziere a intrărilor *NK*, cu *NK(ku)* întârzierea de la intrarea *ku* la oricare din ieşiri se pune *DKX= [D,K,X,NK]* cu *NK* un vector linie de lungime egală cu numărul intrărilor. Dacă se specifică *NK*, atunci valoarea *D* este ignorată .

Prin *default, DKX=[0,1,1].*

3. MODUL DE LUCRU

- Dacă nu este deja creat, se creează un director/folder de lucru.
- Se creează un sistem dinamic cu una sau mai multe intrări, cu una sau mai multe ieşiri. Secvenţa următoare, utilizată şi în alte lucrări este tipică pentru generarea funcţiei de transfer pentru unul din canalele (1,1, de pildă) intrare-ieşire.

```
sisc11=tf([1 2],[1 2 5])
sisd11=c2d(sisc,0.01,'zoh')
```

Ea poate fi utilizată, cu modificări de coeficienţi potrivite, şi la generarea altor funcţii de transfer, pentru alte canale.

- Se aplică sistemului intrări, preferabil sub forma unor secvenţe binare aleatoare

```
u1=sign(randn(size(t)));
```

şi se observă ieşirile. Ieşirile se ″corup″ cu secvenţe de zgomot alb

```
sig=0.0001;
y1=y1+sig*randn(size(t));
```

cu *sig* la alegere, cu *t* vectorul de momente echidistante la care se obsevă sistemul, sau cu secvenţe de zgomot colorat. În acest din urmă caz, zgomotul alb este trecut mai întâi printr-un filtru $C(q^{-1})$.
Se constituie astfel matricea Z de date ″experimentale″.

- Se invocă funcţia ***n4sid*** şi se examinează rezultatele identificării pentru structuri uşor modificate şi pentru erori în date (sig) de importanţă mai mare sau mai mică.
- Prin utilizarea funcţiilor **MATLAB** de conversie a modelelor, când este posibil se revine la forma ″funcţie de transfer″ şi se compară cu modelul care a generat datele.
- Sunt recomandate reprezentările grafice în acelaşi spaţiu ale datelor experimentale şi ale răspunsurilor calculate cu modelul/modelele estimate.

LUCRAREA 11

TEHNICI DE IDENTIFICARE DINAMICĂ A PROCESELOR. NOŢIUNI DE BAZĂ PRIVIND IDENTIFICAREA MODELELOR DINAMICE ALE PROCESELOR

1. OBIECTIVELE LUCRĂRII

Lucrarea prezintă principiile de bază ale identificării modelelor dinamice ale proceselor şi principalele tipuri de algoritmi de adaptare parametrică ce intervin în metodele de identificare recursive.

2. BREVIAR TEORETIC

2.1. Metoda celor mai mici pătrate

În cadrul metodei celor mai mici pătrate (MCMMP), sistemul se consideră descris de următoarea ecuaţie cu diferenţe

$$A\left(q^{-1}\right)y(t)=B\left(q^{-1}\right)u(t)+e(t) \tag{11.1}$$

unde:

$u(t)$ - mărimea de intrare;

$y(t)$ - mărimea de ieşire;

$e(t)$ - zgomotul alb de medie zero şi dispersie λ^2;

q^{-1} - operator de întârziere.

Modelul se consideră descris de o ecuaţie cu diferenţe care are aceeaşi structură cu ecuaţia de mai sus.

$$\overline{A}\left(q^{-1}\right)y(t)=\overline{B}\left(q^{-1}\right)u(t)+\varepsilon(t) \tag{11.2}$$

unde:

$u(t)$ - mărimea de intrare;

$y(t)$ - mărimea de ieşire;

$\varepsilon(t)$ - reziduul modelului;

q^{-1} - operator de întârziere;

$$\overline{A}\left(q^{-1}\right)=1+\overline{a}_1q^{-1}+\ldots+\overline{a}_{n\hat{a}}q^{-n\hat{a}} \tag{11.3}$$

$$\overline{B}\left(q^{-1}\right)=1+\overline{b}_1q^{-1}+\ldots+\overline{b}_{n\hat{b}}q^{-n\hat{b}} \tag{11.4}$$

Se fac următoarele notaţii:

$$\overline{\theta}=\left[\overline{a}_1\ldots\overline{a}_{n\hat{a}}\quad\overline{b}_1\ldots\overline{b}_{n\hat{b}}\right]^T \tag{11.5}$$

$$\overline{\varphi}(t)=\left[-y(t-1)\ldots-y(t-n\hat{a})\quad u(t-1)\ldots u\left(t-n\hat{b}\right)\right]^T \tag{11.6}$$

Cu aceste notaţii, mărimea de ieşire dată de model este

$$y_m(t)=\varphi^{-T}(t)\overline{\theta}+\varepsilon(t) \tag{11.7}$$

Având disponibilă structura modelului (na,nb) se impune condiţia ca media pătratică a erorii de predicţie să fie minimă. Estimaţia celor mai mici pătrate a lui $\overline{\theta}$, bazată pe n date, este prin definiţie

$$\hat{\theta}=\arg\min_{\overline{\theta}}V\left(\overline{\theta}\right), \tag{11.8}$$

unde

$$V\left(\overline{\theta}\right)=\sum_{i=1}^{n}\left[y(t)-\varphi^{-T}(t)\overline{\theta}\right]^2 \tag{11.9}$$

Din condiţia de mai sus rezultă

$$\hat{\theta} = \left(\sum_{i=1}^{N} \overline{\varphi}(t) \overline{\varphi}^{T}(t) \right)^{-1} \sum_{i=1}^{N} \overline{\varphi}(t) y(t) \tag{11.10}$$

In continuare, vor fi enunţate câteva rezultate referitoare la existenţa inversei matricei

$$V_{\theta\theta} = \sum_{i=1}^{N} \overline{\varphi}(t) \overline{\varphi}^{T}(t), \tag{11.11}$$

şi a consistenţei estimatorului $\hat{\theta}$.

Problema consistenţei estimatorului şi a existenţei inversei matricei de mai sus este strâns corelată cu persistenţa semnalului de intrare u. Dacă se notează

$\theta = \left[\overline{a}_1 \ldots \overline{a}_{na} \quad \overline{b}_1 \ldots \overline{b}_{nb} \right]^T$ - parametrii modelului,

iar cu

$\varphi(t) = \left[-y(t-1) \ldots -y(t-n_a) \quad u(t-1) \ldots u(t-n_b) \right]^T$ - vectorul care conţine *istoria* procesului (intrările şi ieşirile anterioare), atunci ecuaţia principală devine

$$y(t) = \varphi^T(t)\theta + \varepsilon(t) \tag{11.12}$$

Modelul va fi descris printr-o ecuaţie de aceeaşi formă

$$y_m(t) = \overline{\varphi}^T(t)\overline{\theta} + \varepsilon(t), \tag{11.13}$$

unde

$\overline{\theta}$ - parametrii estimaţi, iar

$$\overline{\varphi}(t) = \left[-y(t-1) \ldots -y\left(t-\overline{n_a}\right) \quad u(t-1) \ldots u\left(t-\overline{n_b}\right) \right]^T \tag{11.14}$$

Estimarea parametrilor modelului $\left(\overline{\theta}\right)$ presupune în primul rând stabilirea $\overline{n_a}$, $\overline{n_b}$ şi apoi determinare vectorului θ pe baza datelor de intrare şi ieşire.

De fapt, esenţa metodei constă în a presupune faptul că modelul este determinist

$$y_m(t)=\overline{\varphi}^T(t)\overline{\theta}, \tag{11.15}$$

situaţie în care $\hat{\theta}$ se calculează impunându-se următoarea condiţie

$$\hat{\theta}=\arg\min_{\overline{\theta}}\frac{1}{N}\sum_{t=1}^{N}\left(y(t)-y_m(t)\right)^2=\arg\min_{\overline{\theta}}\frac{1}{N}\sum_{t=1}^{N}\left(y(t)-\overline{\varphi}^T(t)\overline{\theta}\right)^2 \tag{11.16}$$

Expresia explicită a lui $\hat{\theta}$ se obţine din condiţia de anulare a gradientului funcţiei criteriu

$$V_{\overline{\theta}}\left(\hat{\theta}\right)=0,\ \frac{\partial V}{\partial\overline{\theta}}\Big|_{\overline{\theta}=\hat{\theta}}=-2\frac{1}{N}\sum_{t=1}^{N}\overline{\varphi}(t)\left(y(t)-\overline{\varphi}^T(t)\hat{\theta}\right)^2=0\ \Rightarrow$$

$$\sum_{t=1}^{N}\overline{\varphi}(t)y(t)=\sum_{t=1}^{N}\overline{\varphi}(t)\overline{\varphi}^T(t)\hat{\theta}\ \Rightarrow \tag{11.17}$$

$$\hat{\theta}=\left[\sum_{t=1}^{N}\overline{\varphi}(t)\overline{\varphi}^T(t)\right]^{-1}\sum_{t=1}^{N}\overline{\varphi}(t)y(t)$$

Dacă se notează cu

$$Y=\left[y(1)\ldots y(N)\right]^T,\ \Phi^T=\left[\overline{\varphi}(1)\ldots\overline{\varphi}(N)\right], \tag{11.18}$$

atunci se obţine

$$\Phi^T\Phi=\sum_{t=1}^{N}\overline{\varphi}(t)\overline{\varphi}^T(t), \tag{11.19}$$

$$\Phi^TY=\left[\overline{\varphi}(1)\ldots\overline{\varphi}(N)\right]\begin{bmatrix}y(1)\\ \vdots\\ y(N)\end{bmatrix}=\sum_{t=1}^{N}\overline{\varphi}(t)y(t), \tag{11.20}$$

$$Y=\Phi\theta+\varepsilon. \tag{11.21}$$

Cu aceste notaţii estimatorul se poate scrie sub forma următoare

$$\hat{\theta} = \left[\Phi^T \Phi\right]^{-1} \Phi^T Y \tag{11.22}$$

Estimatorul dat prin relaţia de mai sus reprezintă *estimatorul celor mai mici pătrate* care s-a obţinut pe baza datelor de intrare $u(1),\ldots,u(N)$ şi a celor de ieşire $y(1),\ldots,y(N)$.

2.2. Metoda variabilei instrumentale

Metoda variabilei instrumentale poate fi privită ca o generalizare a metodei celor mai mici pătrate care furnizează numai partea deterministă a modelului.

Fie sistemul descris de următoarea ecuaţie cu diferenţe

$$A\left(q^{-1}\right) y(t) = B\left(q^{-1}\right) u(t) + \varepsilon(t), \tag{11.23}$$

sau de ecuaţia

$$y(t) = \varphi^T(t) \cdot \theta + \varepsilon(t) \tag{11.24}$$

unde s-a notat cu

$$\varphi(t) = \left[-y(t-1)\ldots -y(t-n_{na}) \quad u(t-1)\ldots u(t-n_{nb})\right]^T$$

$$\theta = \left[a_1 \ldots a_{na} \quad b_1 \ldots b_{nb}\right]^T. \tag{11.25}$$

Estimaţia se poate obţine şi euristic înmulţindu-se relaţia (11.25) la stânga cu $\varphi(t)$

$$\varphi(t) \cdot y(t) = \varphi(t) \cdot \varphi^T(t) \cdot \theta + \varphi(t) \cdot \varepsilon(t)$$

$$\sum_{t=1}^{N} \varphi(t) \cdot y(t) = \sum_{t=1}^{N} \varphi(t) \cdot \varphi^T(t) \cdot \theta + \sum_{t=1}^{N} \varphi(t) \cdot \varepsilon(t) \tag{11.26}$$

$$\hat{\theta}=\left[\sum_{t=1}^{N}\varphi(t)\cdot\varphi^{T}(t)\right]^{-1}\cdot\sum_{t=1}^{N}\varphi(t)\cdot y(t)$$

În această relaţie s-a neglijat termenul

$$\left[\sum_{t=1}^{N}\varphi(t)\cdot\varphi^{T}(t)\right]^{-1}\cdot\sum_{t=1}^{N}\varphi(t)\cdot\varepsilon(t) \tag{11.27}$$

Dacă φ şi ε sunt necorelate, atunci termenul de mai sus care s-a neglijat în relaţia respectivă este nesemnificativ, ceea ce înseamnă ca y şi ε respectiv u şi ε sunt necorelate.

Observaţie
În cazul acestei estimări s-a presupus structura modelului identică cu cea a sistemului. Aspectul care intervine în analiza care urmează se referă la faptul că u şi ε sunt necorelate, dar y şi ε sunt corelate deoarece $y=\varphi^{T}\cdot\theta+\varepsilon$, deci ε se adaugă la ieşire. Pornind de la această constatare se înmulţeşte relaţia care descrie sistemul (a doua din 11.26) cu $z(t)$ format numai din valori ale lui $u(t)$, situaţie în care $\sum_{t=1}^{N}z(t)\cdot\varepsilon(t)$ se poate neglija deoarece $u(t)$ şi $\varepsilon(t)$ sunt necorelate.

Cu aceste observaţii relaţia devine

$$\sum_{t=1}^{N}z(t)\cdot\varepsilon(t)=\left(\sum_{t=1}^{N}z(t)\cdot\varphi^{T}(t)\right)\hat{\theta} \tag{11.28}$$

unde $\hat{\theta}$ se numeşte **estimaţie de variabilă instrumentală**, iar $z(t)$ este **vector de variabilă instrumentală** care nu are o semnificaţie fizică, constituind doar un instrument de lucru.

Dacă se consideră $\dim(z)=\dim(\theta)=n_a+n_b=n_\theta$, atunci

$$\hat{\theta}=\left(\sum_{t=1}^{N} z(t)\cdot\varphi^{T}(t)\right)^{-1}\sum_{t=1}^{N} z(t)\cdot y(t) \tag{11.29}$$

În relaţia anterioară s-a considerat faptul că există inversa matricei

$$\left(\sum_{t=1}^{N} z(t)\cdot\varphi^{T}(t)\right) \tag{11.30}$$

Un vector variabilă instrumentală frecvent utilizat în conjuncţie cu relaţia precedentă este şi vectorul următor

$$z(t)=F\left(q^{-1}\right)\cdot\left[u(t-1)\ldots u(t-n_{\theta})\right]^{T} \tag{11.31}$$

unde $F\left(q^{-1}\right)$ este un filtru stabil.

Dacă nu există informaţii despre sistem, atunci se consideră $F\left(q^{-1}\right)=1$. Dacă sunt disponibile informaţii (de exemplu se cunosc estimaţiile iniţiale $\tilde{A}$, $\tilde{B}$ pentru polinoamele A şi B), atunci se poate alege

$$F\left(q^{-1}\right)=1/\tilde{A}\left(q^{-1}\right) \tag{11.32}$$

Introducerea lui $F\left(q^{-1}\right)$ are o mare importanţă în ceea ce priveşte precizia şi stabilitatea numerică a algoritmului de identificare.

O transformare liniară a vectorului $z(t)$ nu afectează estimaţia. Pentru a demonstra această afirmaţie se consideră următoarea transformare liniară şi nesingulară a lui $z(t)$

$$\tilde{z}(t)=S\cdot z(t) \tag{11.33}$$

unde S este o matrice de dimensiune $\left(n_{\theta}\times n_{\theta}\right)$.

Utilizându-se transformarea de mai sus, relaţia lui $\hat{\theta}$ devine

$$\tilde{\theta}=\left(\sum_{t=1}^{N}\tilde{z}(t)\cdot\varphi^{T}(t)\right)^{-1}\sum_{t=1}^{N}\tilde{z}(t)\cdot y(t)=\left(S\cdot\sum_{t=1}^{N}z(t)\cdot\varphi^{T}(t)\right)^{-1}\left(S\cdot\sum_{t=1}^{N}z(t)\cdot y(t)\right)=$$

$$=\left(\sum_{t=1}^{N}z(t)\cdot\varphi^{T}(t)\right)^{-1}\left(\sum_{t=1}^{N}z(t)\cdot y(t)\right)=\hat{\theta} \qquad (11.34)$$

ceea ce justifică faptul că transformarea respectivă nu afectează estimaţia.

Pentru a pune în evidenţă anumite aspecte ale estimatorului de variabilă instrumentală se consideră transformarea următoare

$$\tilde{z}(t)=S\cdot z(t)=\begin{bmatrix}-\dfrac{\tilde{B}(q^{-1})}{\tilde{A}(q^{-1})}u(t-1)\\ \vdots\\ -\dfrac{\tilde{B}(q^{-1})}{\tilde{A}(q^{-1})}u(t-n_a)\\ u(t-1)\\ \vdots\\ u(t-n_b)\end{bmatrix}=\begin{bmatrix}-\tilde{y}(t-1)\\ \vdots\\ -\tilde{y}(t-n_a)\\ u(t-1)\\ \vdots\\ u(t-n_b)\end{bmatrix}=\begin{bmatrix}-\tilde{z}(t-1)\\ \vdots\\ -\tilde{z}(t-n_b)\\ u(t-1)\\ \vdots\\ u(t-n_b)\end{bmatrix}. \qquad (11.35)$$

În relaţia de mai sus s-a considerat $\tilde{A}(q^{-1})\tilde{y}(t)=\tilde{B}(q^{-1})u(t)+\tilde{\varepsilon}(t)$, unde s-a neglijat reziduul aleator $\tilde{\varepsilon}(t)$, iar $\tilde{A}(q^{-1})$ şi $\tilde{B}(q^{-1})$ reprezintă estimaţii iniţiale (aproximative) ale părţii deterministe a modelului.

Se poate arăta faptul că dacă $\tilde{A}(q^{-1})$ şi $\tilde{B}(q^{-1})$ sunt prime, atunci mărimea S este nesingulară. Utilizându-se transformarea de mai sus se

poate afirma că pentru aproximaţii bune a părţii deterministe $\left(\tilde{A};\tilde{B}\cong A;B\right)$ şi pentru un N suficient de mare are loc următoarea relaţie

$$\sum_{t=1}^{N}\tilde{z}(t)\cdot\varphi^{T}(t)\cong\sum_{t=1}^{N}\tilde{z}(t)\cdot\tilde{z}^{T}(t) \tag{11.36}$$

deci matricea care se inversează tinde, pe măsură ce estimaţiilor $\tilde{A}$, $\tilde{B}$ se corectează, la o matrice simetrică, pozitiv definită şi mai bine condiţionată decât o matrice nesimetrică şi indefinită. În acest caz estimatorul variabilă instrumentală obţinut este mai precis, ceea ce justifică calculele efectuate cu transformarea $\tilde{z}(t)=S\cdot z(t)$.

O problemă care se analizează în cele ce urmează este şi consistenţa estimatorului de variabilă instrumentală. Se consideră sistemul stochastic monovariabil (S) definit prin ecuaţia următoare

$$A\left(q^{-1}\right)y(t)=B\left(q^{-1}\right)u(t)+\nu(t) \tag{11.37}$$

iar modelul descris prin ecuaţia

$$\overline{A}\left(q^{-1}\right)y(t)=\overline{B}\left(q^{-1}\right)u(t)+\overline{\nu}(t) \tag{11.38}$$

unde

$\nu(t)=H\left(q^{-1}\right)e(t)$, $\overline{\nu}(t)=\overline{H}\left(q^{-1}\right)\varepsilon(t)$,

$e(t)$ - zgomot alb,

$\varepsilon(t)$ - reziduurile modelului.

După cum se ştie $H\left(q^{-1}\right)$ şi $\overline{H}\left(q^{-1}\right)$ reprezintă modelul zgomotului, iar în cele ce urmează se consideră de forma următoare

$$H\left(q^{-1}\right)=1+c_1q^{-1}+\ldots c_{nc}q^{-nc} \tag{11.39}$$

$$\overline{H}\left(q^{-1}\right)=1+\overline{c}_1q^{-1}+\ldots \overline{c}_{n\overline{c}}q^{-nc} \tag{11.40}$$

Se defineşte vectorul variabilă instrumentală de forma

$$z(t)=\left[-y\left(t-1-\hat{n}_c\right)\ldots-y\left(t-n_c-\hat{n}_c\right)\quad u\left(t-1\right)\ldots u\left(t-\hat{n}_b\right)\right]^T \tag{11.41}$$

În aceste condiţii estimaţia $\hat{\theta}$ este dată de relaţia următoare

$$\hat{\theta}=\theta+\left[\frac{1}{N}\sum_{t=1}^{N}z(t)\cdot\varphi^T(t)\right]^{-1}\cdot\left(\sum_{t=1}^{N}z(t)\cdot\nu(t)\right) \tag{11.42}$$

Termenul $\sum_{t=1}^{N}z(t)\cdot\nu(t)\to 0$ din relaţia anterioară tinde la zero daca N tinde la infinit, deoarece $z(t)$ nu mai este corelat cu $\nu(t)$. Aceasta se justifică prin faptul că $\nu(t)$ conţine eşantioane până la momentul $\left(t-\hat{n}_c\right)$, iar $z(t)$ conţine eşantioanele $-y\left(t-1-\hat{n}_c\right)\ldots-y\left(t-n_c-\hat{n}_c\right)$ care nu mai sunt corelate cu $\left\{\varepsilon\left(t-i\right)_{i=1,n_c}\right\}$.

În cele ce urmează se face următoarea notaţie

$$\tilde{n}=\min\left(\hat{n}_a-n_a,\hat{n}_b-n_b,\hat{n}_c-n_c\right) \tag{11.43}$$

necesară estimării teoremei de mai jos.

În continuare este prezentat algoritmul metodei variabilei instrumentală pentru cazul când vectorul $\tilde{z}$ este de forma

$$\begin{aligned}\tilde{z}&=\left[\tilde{z}\left(t-1\right)\ldots\tilde{z}\left(t-\bar{n}_a\right)\quad\tilde{z}\left(t-\bar{n}_a-1\right)\ldots\tilde{z}\left(t-\bar{n}_a-\bar{n}_b\right)\right]^T=\\&=\left[\tilde{z}\left(t-1\right)\ldots\tilde{z}\left(t-\bar{n}_a\right)\quad u\left(t-1\right)\ldots u\left(t-\bar{n}_b\right)\right]^T\end{aligned} \tag{11.44}$$

unde

$$\tilde{z}(t-i)=\frac{\tilde{B}\left(q^{-1}\right)}{\tilde{A}\left(q^{-1}\right)}\cdot u(t-1) \quad i=\overline{1,\bar{n}_a} \tag{11.45}$$

Înregistrările obţinute de la proces se presupun cunoscute şi se notează cu $\left\{u(t),y(t)\right\}_{t=\overline{1,N}}$.

Algoritmul metodei de variabilă instrumentală

Pas 1: Se determină $\hat{A}^{\circ}$ şi $\tilde{B}^{\circ}$ utilizând metoda celor mai mici pătrate.

Pas 2: Se iniţializează contorul $i=1$

Pas 3: Se determină estimaţia de variabilă instrumentală

$$\hat{\theta}^i=\left(\sum_{t=1}^{N}\tilde{z}(t)\cdot\varphi^T(t)\right)^{-1}\cdot\sum_{t=1}^{N}\tilde{z}(t)\cdot y(t) \tag{11.46}$$

unde

$$\hat{\theta}^i==\left[\hat{a}_1^i\ldots\hat{a}_{na}^i \quad \hat{b}_1^i\ldots\hat{b}_{nb}^i\right]^T \tag{11.47}$$

$$\tilde{z}^i=\left[-\frac{\hat{B}^{i-1}\left(q^{-1}\right)}{\hat{A}^{i-1}\left(q^{-1}\right)}\cdot u(t-1)\ldots-\frac{\hat{B}^{i-1}\left(q^{-1}\right)}{\hat{A}^{i-1}\left(q^{-1}\right)}\cdot u\left(t-\hat{n}_a\right) \quad u(t-1)\ldots u\left(t-\hat{n}_b\right)\right]^T \tag{11.48}$$

$$\varphi(t)=\left[-y(t-1)\ldots-y\left(t-\hat{n}_a\right) \quad u(t-1)\ldots u\left(t-\hat{n}_b\right)\right]^T \tag{11.49}$$

iar $\hat{B}^{i-1}$, $\hat{A}^{i-1}$ reprezintă estimaţiile obţinute pentru valoarea contorului egală cu $(i-1)$.

Pas 4: Se incrementează contorul $i=i+1$;

Pas 5: Dacă $M\left[z(t)\cdot\nu(t)\right]\le\varepsilon_{impuls}$, atunci – STOP, altfel – continuă;

Pas 6: Dacă $i < M_{impuls}$ atunci – salt la Pas3, altfel STOP.

După cum se observă algoritmul se opreşte dacă este îndeplinită una din următoarele condiţii de STOP:

- $M\left[z(t)\cdot\nu(t)\right] \le \varepsilon_{impuls}$, ceea ce înseamnă că vectorul estimatelor $\hat{\theta}$ s-a obţinut cu o precizie bună;
- $i < M_{impuls}$ ceea ce înseamnă că după parcurgerea a unui număr M_{impuls} de iteraţii nu s-a îndeplinit criteriul anterior.

3. Modul de lucru

- Dacă nu este deja creat, se creează un director/folder de lucru.;
- Se activează platforma de lucru **MATLAB**;
- Se consideră modelul descris de următoarea ecuaţie cu diferenţe

$$y(t)-1.5y(t-1)+0.7y(t-2)=u(t-1)+0.5u(t-2)+e(t)$$

- Se elaborează programul care implementează etapa de estimare din cadrul algoritmului MCMMP;
- Se elaborează programul care implementează algoritmul metodei variabilei instrumentale;
- Se analizează comparativ rezultatele obţinute în urma aplicării celor două metode de identificare.

LUCRAREA 12

IDENTIFICAREA EXPERIMENTALĂ A UNUI SISTEM ALCĂTUIT DIN DOUĂ VASE ÎN CASCADĂ

1. OBIECTIVELE LUCRĂRII

Determinarea unui model matematic pe baza unor date experimentale obţinute de la un sistem dinamic. Se au în vedere următoarele metode de identificare: *metoda celor mai mici pătrate liniare şi neliniare, metoda verosimilităţii maxime şi metodele recursive.*

2. BREVIAR TEORETIC

Modelul matematic dinamic al procesului de acumulare într-un vas.

Pentru oricare din vasele din figura 12.1, în regim staţionar este valabilă relaţia

$$Q_i = Q_e \tag{12.1}$$

unde Q_i şi Q_e sunt debitele lichidului la intrarea, respectiv la ieşirea din vas. Este evident că în acest caz nivelul din vas este constant, *H*.

Pentru cazul în care $Q_i \neq Q_e$, are loc un proces de acumulare a lichidului în vas, iar zestrea de lichid a acestuia variază în timp, astfel încât se poate scrie succesiv:

Acumulare=Intrare-Iesire

$$A\Delta H = Q_i \Delta t - Q_e \Delta t$$
$$A\frac{dH}{dt} = Q_i - Q_e \tag{12.2}$$

unde A este aria transversală a vasului.

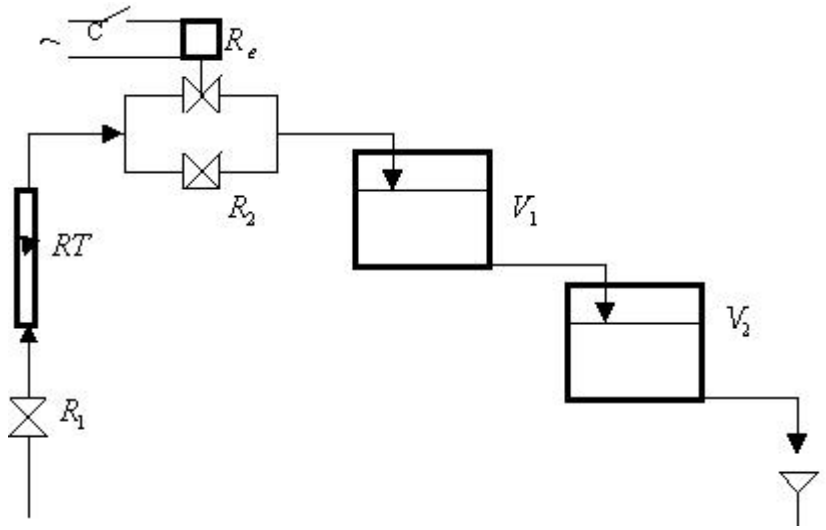


Fig.12.1. Schema instalaţiei experimentale alcătuită din două vase în cascadă

R_1 - rotametru, R_e - robinet cu servomotor electromagnetic, V_1, V_2 - vase de acumulare ; C - contact.

Deoarece scurgerea din vas este liberă, debitul Q_e se poate exprima cu ajutorul relaţiei

$$Q_e = \alpha A_r \sqrt{2gH} \tag{12.3}$$

unde A_r este aria secţiunii minime de trecere prin robinetul instalat pe calea de evacuare a lichidului, α - un coeficient de debit, g -acceleraţia gravitaţiei. Rezultă ecuaţia

$$A\frac{dH}{dt} = Q_i - \alpha A_r \sqrt{2gH} \tag{12.4}$$

care în regim stationar are partea stângă nulă şi în care atât înălţimea H cât şi debitul Q_i se înlocuiesc cu valorile staţionare H_0 şi Q_{i0} . Dezvoltând funcţia $\sqrt{H}$ în serie Taylor în jurul punctului $H = H_0$ şi reţinând numai primii doi termeni ai dezvoltării rezultă

$$\sqrt{H} \approx \sqrt{H_0} + \frac{1}{2\sqrt{H_0}}(H - H_0) = \sqrt{H_0} + \frac{1}{2\sqrt{H_0}}\Delta H \tag{12.5}$$

Prin înlocuire în ecuaţia diferenţială de mai sus se obţine

$$\frac{d\Delta H}{dt} = a\Delta H + b\Delta Q_i \tag{12.6}$$

In relaţiile ultime s-au notat $\Delta H = H - H_0$, $\Delta Q_i = Q_i - Q_{i0}$ abaterile respectivelor mărimi de la valorile care corespund regimului staţionar şi

$$a = -\frac{\alpha A_r \sqrt{2g}}{2A\sqrt{H_0}}, \; b = \frac{1}{A} \tag{12.7}$$

Ecuaţia diferenţială în abateri de la regimul staţionar este varianta liniarizată a ecuaţiei diferenţiale generale scrise mai devreme, care este neliniară. Varianta ultimă este valabilă numai în apropierea punctului staţionar reprezentat de H_0 si Q_{i0}.

Modelul matematic dinamic pentru două vase în cascadă

Pentru vasul al doilea din figura 12.1 se poate scrie, de asemenea, ecuaţia diferenţială generală

$$A\frac{dH}{dt} = Q_i - Q_e \tag{12.8}$$

şi pentru individualizarea cazurilor mărimile pot primi un indice suplimentar, 1 sau 2, întocmai cum volumele celor două vase sunt notate în figură cu V_1 si V_2 .
Desigur, $Q_{e1} = Q_{i2.}$

Intrucât scurgerea din vasul al doilea este, de asemenea, liberă, ecuaţia pentru vasul al doilea este

$$A_2 \frac{dH_2}{dt} = \alpha_1 A_{\nu 1} \sqrt{2gH_1} - \alpha_2 A_{\nu 2} \sqrt{2gH_2} \tag{12.9}$$

şi este o ecuaţie neliniară. Un procedeu similar de exprimare liniară a radicalilor în jurul unor valori de regim staţionar H_{10} şi H_{20} conduce la o ecuaţie diferenţială liniară în ΔH_2 în care apare şi ΔH_1

$$\frac{d\Delta H_2}{dt} = c\Delta H_1 + e\Delta H_2 \tag{12.10}$$

cu coeficienţii c şi e care se pot exprima în funcţie de geometria vaselor V_1 şi V_2 şi de secţiunile de trecere minime ale căilor de evacuare a lichidului din fiecare vas

$$c = \frac{\alpha_1 A_{\nu 1} \sqrt{2g}}{2A_2 \sqrt{H_{10}}}, \quad e = -\frac{\alpha_2 A_{\nu 2} \sqrt{2g}}{2A_2 \sqrt{H_{20}}} \tag{12.11}$$

Modelul liniarizat al ansamblului celor două vase este, aşadar,

$$\begin{aligned} \frac{d\Delta H_1}{dt} &= a\Delta H_1 + b\Delta Q_{i1} \\ \frac{d\Delta H_2}{dt} &= c\Delta H_1 + e\Delta H_2 \end{aligned} \tag{12.12}$$

cu coeficienţii a şi b adaptaţi corespunzător vasului V_1 .

In cazul în care evacuarea lichidului din oricare din vase se face prin conducte lungi, atunci modelul matematic al vaselor se modifică. Sistemul îsi modifică ordinul deoarece pentru fiecare vas se mai adaugă o ecuaţie

$$\frac{dQ_e}{dt} = \frac{\pi d^2}{4\rho l}\left(\rho g H - \frac{8 f \rho l Q_e |Q_e|}{\pi^2 d^5}\right) \tag{12.13}$$

care exprimă faptul că masa de lichid din conductă are o dinamică proprie de mişcare sub combinaţia de forţe reprezentată de presiunile diferite de la capetele conductei şi de frecarea vâscoasă a lichidului de el însuşi. Factorul de frecare între straturi adiacente de lichid în mişcare *f* este adimensional. Conducta de evacuare are diametrul *d* şi lungimea *l*. Ecuaţia este de luat în considerare ori de câte ori termenul al doilea din paranteză este important comparativ cu primul termen. O valoare a factorului de frecare pentru apă, care corespunde vâscozităţii ei la temperaturi ambiante uzuale, este $f = 2{,}509.10^{-1}$.

Primul din cele două vase din instalaţia experimentală este un sistem dinamic de ordinul întâi. Modelul matematic în varianta lui liniarizată conduce prin integrarea ecuaţiei diferenţiale-model la următorul răspuns la saltul treaptă în debitul Q_{1i} care alimentează vasul

$$\Delta h_1(t) = A(1 - e^{Bt}) \tag{12.14}$$

Sistemul de două vase considerat liniar este de ordinul al doilea. Variaţia cotei în vasul prim rămâne aceeaşi ca în expresia de mai sus. Variaţia cotei în vasul al doilea are expresia generală

$$\Delta h_2(t) = D[1 - E e^{Bt} + (E - 1)e^{Ct}] \tag{12.15}$$

cu *E* dependent de *B* şi *C*, respectiv $E = -C/(B - C)$, ceea ce aduce expresia ultimă la forma

$$\Delta h_2(t) = D[1 + \frac{C}{B - C} e^{Bt} - \frac{B}{B - C} e^{Ct}] \tag{12.16}$$

Ambele expresii se referă la răspunsul sistemului la aplicarea unei modificări treaptă a debitului de lichid Q_{1i} care alimentează primul vas.

Metoda celor mai mici pătrate liniare este utilizată ca metodă de estimare a parametrilor atunci când modelul este liniar sau poate fi liniarizat în raport cu parametrii săi. Se disting două variante ale acestei metode: fără ponderi sau cu ponderi. In ambele variante se caută valorile parametrilor modelului când se cunosc rezultatele observării experimentale a uneia sau mai multor variabile dependente

(ieşiri), în condiţii de bună stăpânire a variabilelor independente (intrări) ale sistemului modelat. Acest lucru semnifică faptul că variabilele independente sunt măsurate/modificate/menţinute cu o precizie net mai bună decât cele dependente, astfel încât valorile primelor se pot considera cunoscute aproape exact, pe când valorile celor din urmă sunt afectate de erori care nu pot fi trecute cu vederea. In cazul în care erorile de observare a variabilei (variabilelor) dependente sunt aceleaşi pe tot domeniul lor de valori, se aplică metoda celor mai mici pătrate fără ponderi, ceea ce rezidă în rezolvarea sistemului de ecuaţii care rezultă din condiţiile de minim al sumei de pătrate

$$S = \sum_{k=1}^{N} (y_{ok} - y_{ck})^2 = (y_o - y_c)^T (y_o - y_c) \tag{12.17}$$

exprimată şi ca o sumă de termeni reprezentând diferenţele experiment-model pentru fiecare k = 1, 2, ..., N din cele N observaţii/experienţe, dar şi în funcţie de vectorii observaţiilor y_o şi valorilor calculate din model y_c .

Dacă erorile care afectează ieşirile sunt diferite pe întinderea domeniului de valori, se aplică metoda celor mai mici pătrate ponderate, adică se rezolvă sistemul de ecuaţii care rezultă din condiţiile de minim al sumei de pătrate

$$S = \sum_{k=1}^{N} q_k (y_{ok} - y_{ck})^2 = (y_o - y_c)^T Q (y_o - y_c) \tag{12.18}$$

cu ponderile q_k (k = 1, 2, ..., N) conţinute explicit în suma de pătrate sau în matricea diagonală Q. Ponderile sunt de obicei invers proporţionale cu erorile medii pătratice sau cu dispersiile observaţiilor în punctele de indice k. In ambele cazuri minimizarea sumei S dependentă pătratic de parametrii modelului se reduce la a rezolva un sistem normal de ecuaţii liniare cu necunoscutele tocmai parametrii de estimat.

Metoda celor mai mici pătrate neliniare este analogă celei liniare: aceleaşi sume de pătrate trebuie minimizate, de la caz la caz cu sau fără ponderi (cu ponderi egale). Suma pătratelor este însă o funcţie mai complicată de parametrii de estimat şi minimizarea ei necesită

utilizarea unor metode speciale, de pildă metodele de gradient sau alte metode încă mai rafinate.

Metoda verosimilităţii maxime se utilizează atunci când erorile de măsurare/menţinere a variabilelor independente nu mai sunt nici ele atât de lipsite de importanţă. Valorile parametrilor maximum verosimile sunt acelea care fac maximă o probabilitate combinată a diferenţelor model-experiment.

Metodele recursive au la bază metodele de mai sus, dar şi altele, caracteristic fiindu-le utilizarea datelor experimentale pe măsura observării lor. Aşadar, ele pot fi utilizate "on-line" pentru estimarea parametrilor modelului asociat unui sistem dinamic dat. Aspectele teoretice ale estimării de parametri recursive au fost date, cel puţin pentru metoda celor mai mici pătrate recomandată a se aplica în lucrarea prezentă, la cursul de *Identificarea sistemelor*.

Prin utilizarea metodelor de estimare enumerate în secţiunea precedentă se vor realiza programe de calcul pentru estimarea din datele experimentale la dispoziţie a parametrilor *A, B, C* şi *D*.

Pentru **metoda verosimilităţii maxime** se presupune că eroarea medie pătratică de citire a timpului este de 1.5 secunde.

Metoda celor mai mici pătrate se aplică aici în varianta neliniară. Intr-adevăr, suma pătratelor abaterilor pentru sistemul de ordinul unu reprezentat de primul vas este

$$s_1(A,B) = \sum_{i=1}^{N} [\Delta h_{1oi}(t) - A(1 - e^{Bt_i})]^2 \tag{12.19}$$

fiind o funcţie neliniară de parametrii *A* şi *B*. La fel, pentru nivelul observat şi calculat la diferite momente în vasul al doilea suma pătratelor diferenţelor

$$s_2(B,C,D) = \sum_{i=1}^{N} [\Delta h_{2oi}(t) - D(1 + \frac{C}{B-C} e^{Bt_i} - \frac{B}{B-C} e^{Ct_i})]^2 \tag{12.20}$$

este o funcţie neliniară în parametri de estimat *B, C* şi *D*. Cele două funcţii sunt măsuri ale distanţei model-experiment şi trebuie minimizate prin stabilirea celor mai potrivite valori ale parametrilor *A, B, C* şi *D*.
Minimizarea funcţiilor obiectiv, cele de mai sus sau altele, se poate obţine pe căi diverse.

Metodele de gradient au în vedere vectorul derivatelor parţiale ale funcţiei de minimizat numit şi gradientul funcţiei. Evaluarea gradientului într-un punct permite stabilirea direcţiei în care funcţia scade/creşte cel mai rapid. O modificare a variabilelor, în cazul de faţă a parametrilor de estimat, proporţională cu valorile derivatelor parţiale, asigură deplasarea pe direcţia gradientului. Variaţii de semn contrar pe fiecare direcţie componentă a gradientului, nu foarte importante, fac ca funcţia să scadă. Evaluări repetate ale gradientului şi deplasarea în spaţiul explorat pe direcţia vectorului gradient conduc în cele din urmă la stabilirea coordonatelor unui extrem (minim). Acesta poate fi în cazul general al funcţiilor cu mai multe extreme un extrem local. Localizarea extremului "cel mai extrem" poate fi uneori problematică. Reluarea calculelor cu un alt punct de pornire oferă o şansă de a atinge un alt minim, care poate fi „cel mai minim" între minimele funcţiei. Sub aspectul rapidităţii cu care este localizat un extrem, metodele de gradient pot fi uneori lente. De aceea s-au pus la punct variante ale metodelor de gradient, care fac căutarea eficientă. pe categorii de funcţii.

Programul recomandat, ***est.pas***, face estimarea de parametri în etape. Mai întâi produce valori pentru parametrii sistemului de ordinul unu constituit de primul din cele două vase în cascadă. Apoi face acelaşi lucru pentru sistemul în ansamblu, menţinând valorile pentru primul vas constante, aşa cum au fost stabilite în prima etapă. Aici se mizează pe relativa independenţă a vasului prim de cel de al doilea, ceea ce este în bună măsură corect. In general, estimarea de parametri trebuie făcută concomitent prin minimizarea sumei celor două funcţii obiectiv tratate aici separat.

Programul propus utilizează metoda celor mai mici pătrate (neliniare) şi metoda verosimilităţii maxime. In cazul din urmă pentru primul vas este obtinută şi o elipsă de încredere, echivalentă intervalului de încredere din cazul unidimensional, elipsă care conţine

în interiorul ei toate perechile de valori ale parametrilor la fel de bune sub aspect statistic ca şi valorile centrale, utilizabile aşadar în egală măsură în modelul dinamic al sistemului. Nu s-a recurs la calculul similar pentru sistemul incluzând al doilea vas deoarece în etapa a doua a calculului sunt estimaţi concomitent trei parametri şi reprezentările grafice sunt mai dificile. Este sugerată completarea programului cu reprezentarea tridimensională a elipsoidului de încredere pentru parametrii estimaţi.
Când datele sunt recoltate în timp real, sunt preferate metodele recursive de estimare a parametrilor. Pe aceleaşi date experimentale, programul ***rec.pas*** execută operaţia de estimare a parametrilor în manieră recursivă. Se admite că sistemul de vase în cascadă este discret sau, mai corect, este descris de un model matematic de tip discret.

3. MODUL DE LUCRU

Determinarea modelului matematic se bazează pe datele experimentale referitoare la evoluţia în timp a cotelor lichidului în cele două vase în cascadă .

Se studiază răspunsul sistemului din figura 12.1 pentru un semnal treaptă aplicat debitului de alimentare a primului vas Q_{i1}.

Desfăşurarea experimentului se realizează după cum urmează:

1. Se aduce procesul într-un regim staţionar caracterizat prin constanţa funcţiilor de timp $Q_{i1}(t)$, $H_1(t)$, $H_2(t)$. Un punct staţionar convenabil este $Q_{i1}(t)$=50 l/oră, $H_1(t)$=75 mm, $H_2(t)$=80 mm. Obţinerea acestor valori se realizează prin manevrarea robinetelor instalaţiei, cu robinetul de acţionare electromagnetică R_e închis.

2. Prin intermediul robinetelor R_e sau R_2 se dă o variaţie treaptă debitului Q_{i1}. Se recomandă o treaptă pozitivă astfel încât Q_{i1} să ajungă la 70-80 l/oră. Se notează evoluţia în timp a nivelurilor H_1 şi H_2 pe indicatoarele de nivel ataşate vaselor. Aceste date vor fi trecute într-un tabel – tabelul 12.1- urmând apoi a fi introduse într-un fişier de date, ***expe.dat.***

Tabelul 12.1

t (s)	Δh_1	Δh_2

In tabel timpul este marcat cu *t*, iar cotele în cele două vase sunt notate cu Δh_1, Δh_2.
Este foarte important ca pe durata regimului tranzitoriu valoarea debitului Q_{i1} să nu se modifice. În acest scop, va fi observat permanent debitul indicat de rotametrul instalaţiei. Micile variaţii se recomandă a fi eliminate prin acţionarea robinetului R_1.

3. Pentru determinarea constantelor de timp, se trasează graficele funcţiilor $H_1(t)$ şi $H_2(t)$.

Identificarea sistemului propus prin determinarea modelului matematic se va realiza prin parcurgerea următoarelor etape:

- Estimarea pe baza datelor experimentale a constantelor de timp ale sistemului cu acumulare de lichid, folosind programul ***est.pas***;
- Estimarea în manieră recursivă, pe baza datelor experimentale, a constantelor de timp ale sistemului, folosind programul ***rec.pas***;
- Compararea valorilor obţinute prin aplicarea diferitelor metode de identificare ;
- Estimarea modelului matematic utilizând alte metode de identificare: metoda erorii de predicţie, metoda variabilei instrumentale, cu elaborarea programelor software de calcul corespunzătoare.

program est;

```
{ Program de estimare a parametrilor }
 uses CRT,graph;

const
dt=15.0;
sir:array [1..18] of string=
('E C U A T I I diferentiale exacte:',
'dVl/dt=FO-al.uVl',
'dV2/dt=al.uVl-a2.uV2',
'Regimul initial (stationar): t=0 => F0=al.uV10=a2.uV20',
'Aproximare   liniart:',
'uV=u(VO+eV)=~uVO.(l+«eV)',
'E C U A T I I liniarizate:',
'deVl/dt=eF-(a I/(2.UV10)).eVI',
'deV2/dt=(al/(2.UV10)).eVI-(a2/ (2.uV20)).eV2',
'S o 1 u t i a   sistemului liniarizat:',
'eVl=A.(l-exp(-al.t)) cu al=al/(2.uVlO)',
'eV2=B+C.exp(-al.t)-(B+C).exp(-a2.t) cu a2=a2/(2.uV20)',
' Sunt de   estimat   cinci parametri: A, B, C, al si a2',
'Date   experimentale   (En fisierul EXPE.DAT',
  'Metoda de estimare:',
 ' Initializati (En PARA. DAT pe o linie A, al pe o a doua linie B, al, C, a2! ')
 Opt:array [1..2] of string=('Cele mai mici patrate',
                                        'Verosimilitate maxima') ;

 type
   deri=array [1..3,1..60] of double;

 var
   f,g:text;
 gdriver,gmode,i,j,k,m,n,nit,opty,x0,yO:integer;
 eps, rol, ro2, t0,xl,x2, yl,y2, yd, yc2 ,u,we: array [1..60] of double;
 sd:deri;
 alf,alfa,alfa0,bet,beta,betaO,gama,gamaO,delta,deltaO,dl,d2,
 pas,s,sO,samare,sarnica,ualfa:double;
 tt,uu,xx,yy,zz:string;
 car:char;
 cov:array [1..3,1..3] of double;

label
  unu;

procedure axe;
 begin
   for i:=0 to 1 do begin
   line(30,400+i,610,400+i);
   line(30-i,50,30-1,400) ;
   line(30-i,50,25-1,55) ;
   line(30-i,50,35-i,55) ;
   line(610,400+1,605,395+i) ;
   line(610,400+i,605,405+i)
   end;
   for i:=0 to 8 do begin
   line(30+70*1,400,30+70*1,397) ;
   str(100*i:3,xx) ;
   outtextxy(30+70*i,410,xx)
   end;
   outtextxy(615,410,'[s]') ;
```

```
    for i:=0 to 3 do begin
    line(30,400-100*1,33,400-100*i) ;
      str(20*i:2,xx) ;
 outtextxy(15,400-100*1,xx)
 end;
 end;

 procedure axel;
   var
     Ix,ly / xO /yO:integer;
   begin
     settextjustifyd, 1) ;
  for i:=0 to 1 do begin
  line(30,400+1,610,400+1);
  line(30-1,50,30-1,400);
  line(30-i,50,25-i,55) ;
  line(30-i,50,35-i,55) ;
  line(610,400+i,605,395+i) ;
  line(610,400+i,605,405+i)
  end;
  for i:=0 to 6 do begin
  line(30+90*1,400,30+90*1,397);
  str(85+5*1:3,xx) ;
  outtextxy(30+90*i,410,xx)
  end;
  outtextxy(615,410,'A [nm]') ;
  for i:=0 to 3 do begin
  line(30,400-100*1,33,400-100*1) ;
  str(1.3+0.l*i:3:l,xx) ;
  outtextxy(15,400-100*i,xx)
  end;
  outtextxy(60,40,'1000.al [1/s]');
  end;

 procedure dubluvas;
var
 alfal,alfa2,ca0,cal,ca2,f0,kl,k2,ro0,rol,ro2,vl,v2:double;
                              kll,kl2,kl3,kl4,k21,k22,k23, k2 4 , past, t, vll, v21: double;
  1:integer;
  function dery(1:integer;vl,v2:double):double;
   begin
   case i of 1: dery:=2.0*beta0*xl[1]*(1.6-sqrt(1.0+vl/xl[1]))
   2: dery:=2.0*beta0*xl[1]*sqrt(1.0+vl/xl[1])
   -2.0*delta0*x2[l]*sqrt(1.0+v2/x2[1]) ;
     end;
  end;
begin
 past:=15;
 vll:=0.0;
 v21:-0.0;
 t:=0.0;
 i:=0;
 repeat
 t:=t+past;
 i:=i+l;
 xO:=30+round(4*t) ;
```

```
vl:-vll;v2:=v21;
kll =past*dery(l,vl,v2) ;
k21 -past*dery(2,vl,v2);
kl2 =past*dery(l,vl+kll/2.0,v2+k21/2.0) ;
k22 =past*dery(2,vl+kll/2.0,v2+k21/2.0) ;
kl3 =past*dery(l,vl+kl2/2.0,v2+k22/2.0) ;
k23 =past*dery(2,vl+kl2/2.0,v2+k22/2.0) ;
kl4 =past*dery(l,vl+kl3,v2+k23) ;
k24 =past*dery(2,vl+kl3,v2+k23) ;
vll =vl+(kll+2.0*kl2+2.0*kl3+kl4)/6.0;
v21 =v2+(k21+2.0*k22+2.0*k23+k241/6.0 ;
ycl[i]:=vll;yc2[i]:=v21;
 until t>780;
 end;

procedure sl(var s:double;var sd:deri;m:integer) ;
var
a:double;
   begin
     3:=0.0;
 for i:=l to 2 do sd [i, 1] : =0 . 0;
 for i:=l to n do begin
 a:-exp(-beta*(i-1)*dt) ;
 yl[i]:=alfa* (1.0-a) ;
 s:=s+sqr(yl[i]-xl[i]+xl[1]) ;
 if m>0 then begin
 sd[l,l]:=sd[l,l]+2.0*(yl[i]-xl[i]+xl[1])* (1.0-a) ;
 sd[2,l]:=sd[2,l]+2.0*(yl[i]-xl[i]+xl[1])*alfa*a*(i-l)*dt
 end
                                end;
 end;

procedure s2(var s:double;var sd:deri;m:integer);
  var
    a,b:double;
 begin
    s:=0.0;
for i:=l to 3 do sd[i,1]:=0.0;
for i:=l to n do begin
a:=exp (-beta*(i-1)*dt) ;
b:=exp(-delta*(i-1)*dt) ;
y2 [i]:=alfa+gama*a-(alfa+gama)*b;
s:=s+sqr(y2[i]-x2[i]+x2[1]);
if m>0 then begin
sd[l,l]:=sd[l,l]+2.0*(y2[i]-x2[i]+x2[1])*(1.0-b) ;
sd[2,l] :=sd[2,l]+2.0*(y2[i]-x2[i]+x2[l] )* (a-b) ;
sd[3,1]:=sd[3,1]+2.0*(y2[i]-x2[i]+x2[l])*(alfa+gama)*(i-1)*dt
end
 end;
end;

procedure s3(var s:double;var sd:deri;m:integer) ;
  var
    a:double;
jx:array [1..3] of double;
  begin
    s:=0.0;
```

```
 for i:=l to n do begin
 a:=exp(-beta*(i-1)*dt) ;
 yl [i]:=alfa* (1.0-a) ;
 eps[i]:-yl[i]-xl[i]+xl(l];
 jx[l]:=alfa*a*beta;
 jx[2]:=-1.0;
 we [i] :=cov[l,l]*sqr(jx[l] )+cov [2 , 2 ] *sqr ( jx [2 ] ) ;
 rol [i] :=abs (eps [i] ) /sqrt (we [i] ) ;
 s : =s+sqr (eps [i] ) /we [i] ;
 if m>0 then begin
 sd[l,i]:=1.0-a;
 sd[2,i]:=alfa*a*(i-1)*dt
 end
  end;
 end;

 procedure s4(var s:double;var sd:deri;m:integer) ;
var
 a,b:double;
 jx:array [1..3] of double;
begin
 s:=0.0;
 for i:=l to n do begin
 a:=exp(-beta*(i-1)*dt) ;
 b:=exp(-delta*(i-1)*dt) ;
 y2 [i]:=alfa+gama*a-(alfa+gama)*b;
 eps[i] :=y2[i]-x2[i]+x2[l] ;
 jx[1]:=-gama*a*beta+(alfa+gama)*b*delta ;
 jx[2] :=-1.0;
 we [i] :=cov [1,1] *sqr (jx[l] ) +cov[3, 3] *sqr (jx[2]);
 ro2 [i] :=abs(eps[i])/sqrt (we [i]) ;
      s:-s+sqr (eps [i])/we [i];
 if m>0 then begin
 sd[l,i]:=1.0-b;
 sd[2,ij:=a-b;
 sd[3,i]:=(-gama*a+(alfa+gama)*b)*(i-1)*dt
 end
                                  end;
 end;

 procedure eli(m:integer);
    var
     btb,btbi,bteb,wa:array [1..3,1..3] of double;
 c,fl0,f20:double;
   begin
 {   restorecrtmode;}
    for i:=l to 3 do for j:=l to 3 do
    begin btb [i, j ] : =0 . 0 ;btbi [i, j ] : =0 . 0,-bteb [i, j ] :=0 . 0 end;
 for i:=l to 3 do btbi [i, i] :=1. 0;
 for j:=1 to m do
 for k:=j to m do begin
 for i:=l to n do btb [j,k]:=btb(j,k]+sd[j,i]*sd[k,i];
 btb[k,j]:=btb[j,k]
                                   end;
 for j : =1 to m do
 for k:=l to m do
 for i:=l to n do bteb[j,k]:=bteb(j,k]+sd[j,i]*we [i]*sd[k,i];
 for i:=l to m do begin
 c:=btb[i,i];
 for j:=i to m do btb[i,j]:=btb[i,j]/c;
 for j:=l to m do btbi[i,j]:=btbi[i,j]/c;
```

```
for k:^! to m do
if koi then begin
c:=btb [k,i];
for j:=i to m do btb[k,j]:=btb[k,j]-c*btb[i,j] ;
for j:=l to m do btbi [k,j]:=btbi[k,j]-c*btbi[i,j]
end;
end;
for j:=l to m do for k:=l to m do btb [j,k]:=0.0;
for j:=l to m do for k:=l to m do
for i:=l to m do btb [j,k]:=btb[j,k]+btbi[j,i]*bteb[i,k];
for j:=l to m do for k:=l to m do wa [j , k] :=0 . 0;
for j:=l to m do for k:=l to m do
for i:=l to m do wa [j,k]:=wa [j,k]+btb[j,i]*btbi[i,k];
for j:=l to m do for k:=l to m do
begin bteb[j,k]:=0.0;if j=k then bteb[j,k]:=1.0 end;
for i:=l to m do begin
c:=wa[i,i];
for j^i to m do wa [ i / j ] : =wa [ i, j ] /c;
for ]:=! to m do bteb[i,j]:=bteb[i,j]/c;
for k:=l to m do
if koi then begin
c: =wa[k, i] ;
for j:=i to m do wa [k,j]:=wa[k,j]-c*wa[i,j];
for j:=l to m do bteb [k,j]:=bteb [k,j]-c*bteb [i,j]
end;
                                end;
writeln(wa[l,l],wa[l,2],wa[2,l],wa[2,2]);
samare:=bteb[l,l]+bteb[2,2];
samica:=bteb[l,l]*bteb[2,2]-bteb [1,2]*bteb[2, 1] ;
samica : =sqrt f sqr (samare) -4.0 ^"s arnica) ;
fl0:=(samare-samica)/2.0;
f20: =(samare+samica)/2.0;
samare:=2.45/sqrtfflO) ;
samica:=2.45/sqrt(f20) ;
f20:--bteb[2,1]/ (bteb [2,2]-flO) ;
ualfa:=57.297*arctan(f20) ;
writein(samare,samica,ualfa) ;
axel;
setcolor(12)
line(30+round(18*(alfa0-2*samare-85)),400-round(1000*(1000*beta0-l.3))
,30+round(18*(alfa0+2*samare-85)),400-round(1000*(1000*beta0-l.3)))
line(30+round(18*(alfaO-85)),400-round(1000*(1000*beta0-2000*samica-l.3))
,30+round(18*(alfaO-85)),400-round(1000*(1000*beta0+2000*samica-l.3))),
set color (15) ;
   ellipse(30+round(18*(alfaO-85) )
                ,400-round(1000*(1000*beta0-l.3) ), 0,360
                ,round (18*samare),round(l.e6*samica)) ;
 outtextxy(300,450,'Pentru continuare apasati orice tasta!');
 outtextxy (400,70,'Elipsa de (Encredere (95%) pentru vasul nr.l');
 str(samare:6:2,xx) ;
 str(1000*samica:6:3,yy);
 str (ualfa:6:3,zz);
 outtextxy(400,90,'Semiaxa mare:       '+xx+'  [mm]') ;
 outtextxy (400,100,'lOOOxSemiaxa mict:   '+yy+' [1/s]');
 outtextxy(400,110,'Inclinarea axei mari:'+zz+'0     ');
 car:=readkey;
 res torecrtmode;
  setgraphmode(getgraphmode)
end;

function spl (alfa,beta:double):double;
 var
    suma:double;
i:integer;
 begin
```

```
    suma : =0 . 0;
    for i:=l to n do suma:=suma+sqr(xl[i]-alfa*(1.0-exp(beta*t0[i]))) ;
    spl:=suma
 end;

function sp2(alfa,beta,gama,delta:double):double;
 var
   suma,co:double;
 i:integer;
 begin
   co:=gama/(beta-gama);
   suma:=0.0;
   for i:=l to n do
 suma:=suma+sqr (x2[i]-delta*(1.0-co*u[i]+(co-1.0) *exp(gama*t0 [i]) ) ) ;
   sp2:=suma
 end;

begin
 assign (f,'expe.dat') ;
reset (f) ;
   readin(f,n) ;
for i:=l to n do readin(f,tO[i],xl[i],x2[i]);
close(f);
for i:=2 to n do u[i]:=ln (1.0-(xl[i]-xl[1])/103.5) ;
samare:=0.0;
samica:=0.0;
for i:=2 to n do begin
   samare:=samare+sqr(tO[i]) ;
sarnica:=samica+t0[i]*u[i]
end;
samica:=samica/samare ;
samare:=0.0;
for i:=2 to n do samare:=samare+sqr(samica*t0[i]-u[i]);
write(samica,samare);readin;
assign(f,'cov.dat') ;
reset(t) ;
readin(f,nit);
for i:=l to 3 do for j:=l to 3 do read(f,cov[i,j]);
close(f) ;
assign(g,'para.dat');
reset(g) ;
readin(g,alfa0,beta0);
readin(g,alfa0,beta0,gamaO,deltaO) ;
close(g) ;
s0:=spl(alfa0,beta0) ;
dl:= (sO-spl(1.001*alfa0,beta0))/ (0.001*alfa0) ;
d2:=(sO-spl(alfa0,1.001*beta0))/(0.001*beta0) ;
 write(dl,d2) ;
( readin;
   exit;)
   detectgraph(gdriver,gmode) ;
InitGraph(gdriver,gmode,'');
dubluvas ;
settextjustify(1,1);
outtextxy(500,50,'OPTIUNE ASUPRA METODEI');
rectangle(400,60,600,105) ;
k:=l;
outtextxy(320,420,
   'Selectare: tastele cu săgeti '+#24+#25+'; Activare: Enter.')
repeat
   for i:=l to 2 do begin
   if i^k then setcolor(13) else setcolor(15);
outtextxy(500,60+15*1,opt[i] ) ;
end;
```

```
 car:=readkey;
 if car=#0 then begin
 car:=readkey;
 if car=#80 then k:=k+l;
 if car=#72 then k:=k-l;
 if k<l then k:=2;
 if k>2 then k:=l;
 end;
 until car=#13;
 opty:=k;
 restorecrtmode ;
 setgraphmode(getgraphmode) ;
 settextjustify(0, 1) ;
 for i:=l to 18 do outtextxy(30,40+15*i,sir[i]);
 setcolor (13) ;
 outtextxy(30,280,sir[16]+opt[opty]) ;
 setcolor (15) ;
 settextjustify (1,1) ;
 outtextxy(300,450,'Pentru continuare apasati orice tasta!');
 car:=readkey;
 restorecrtmode;
 setgraphmode(getgraphmode) ;
 settextjustify (1,1) ;
 assign(g,'para.dat');
 reset(g);
 readin(g,alfa0,beta0) ;
 close(g) ;
 randomize;
 axe;
   outtextxy(40,40,'ehl [mm]') ;
    for i:=l to n do begin
      circle(30+round(0.7*t0[i]),400-round(5*(xl[i]-xl[1])),2) ;
   setcolor (11);
   circle(30+round(0.7*(tO[i]+15)),400-round(5*ycl[i]),2) ;
   setcolor(15)
                              end;
 x0:=30;y0:=400;
   alfa:=alfa0;
   beta:=beta0;
   moveto(xO,yO) ;
   case opty of 1: sl(s0,sd,0); 2: s3(s0,sd,0); end;
   for i:=l to n do begin
     x0:=30+round(0.7*(i-1)*15) ;
   y0:=400-round(5*yl[i]) ;
   lineto(x0,y0) ;
   end;
   str(alfa0:8:3,xx);str(betaO:11:8,yy);str (sO:8:4,zz) ;
   outtextxy(300,430,'A='+xx+' al='+yy+' S^'+zz);
   outtextxy(300,450,'Pentru continuare apasati orice tasta!'it-
   car: =readkey;
   setcolor(0) ;
 for i:=l to n do begin
   circle(30+round(0.7*t0[i]),400-round(5*(xl[i]-xl[1])),2) ;
   circle(30+round(0.7 *(t0(i]+15)),400-round(5*yc1[i]),2)
                              end;
 x0:=30;y0;=400;
 alfa:=alfa0;
 beta:=beta0;
 moveto(xO,yO);
 case opty of 1: sl (sO, sd, 0) ; 2: s3(s0,sd,0); end;
for i:=l to n do begin
 x0:=30+round(0.7*(i-1)*15) ;
 y0:=400-round(5*yl[i]) ;
```

```
 lineto(x0,y0);
 end;
 outtextxy(300,450,'Pentru continuare apasati orice tasta!');
 setcolor(15) ;
 k:=0;
repeat
 x0:=30;y0:=400;
 alfa:=alfa0+0.05*(1.0-2.O*random) ;
 beta:=beta0+0.0001*(1.0-2.O*random) ;
 moveto(xO,yO) ;
 case opty of 1: sl(s,sd,0); 2: s3(s,sd,0); end;
   for i:=l to n do begin
     x0:=30+round(0.7*(i-1)*15);
 y0:=400-round(5*yl[i] ) ;
 lineto(xO,yO) ;
 end;
   for i:=l to n do
     circle(30+round(0.7*t0[i]),400-round(5*(xl[i]-xl [ 1])),2);
 delay (100) ;
 setcolor(0) ;
 x0:=30;y0:=400;
 moveto(xO,yO) ;
 case opty of 1: sl(s,sd,0); 2: s3(s,sd,0); end;
   for i:=l to n do begin
     x0:=30+round(0.7*(i-1)*15) ;
 y0:=400-round(5*yl[i]) ;
 lineto (x0,y0) ;
 end;
 setcolor(15);
 if s0>s then begin alfaO:=alfa;beta0:=beta;sO:=s;k:=l end else k:=k+l;
   if k=10*(k div 10) then begin
     if k>0 then
     begin setcolor(0);outtextxy(500,320,xx+yy);setcolor(15) end;
 str(s:10:3,xx);str(k:5,yy);outtextxy(500,320,xx+yy)
 end;
 until k=nit;
 x0:=30;y0:=400;
  moveto(xO,yO) ;
  alfa:=alfa0;beta:=beta0;
  case opty of 1: sl(s,sd,0>; 2: s3(s,sd,0); end;
 for i:=l to n do begin
  x0:=30+round(0.7*(i-1)*15) ;
  y0:=400-round(5*yl[i]) ;
  lineto(x0,y0);
  end;
  alf:=alfa0;bet:=beta0;
  setcolor (0) ,-outtextxy (500, 320,xx+yy) ; setcolor (14) ;
  sir (alfaO: 8: 3,xx) ; str (betaO : 11: 8,yy) ; str (s0 : 8 : 4, zz) ;
  outtextxy(300,440,'A='+xx+' al='+yy+' S='+zz);
  setcolor(15);
 case opty of 1: begin
  str (s0/ (n-2):6:4,xx) ;
  str(n-2:3,yy);
  outtextxy(500,320,'sy='+xx);
  outtextxy(500,335,'Grade de libertate: '+yy);
  str(sqrt(s0/(n-2)):6:3,xx) ;
  outtextxy(500,350,'Abatere medie patratica: '+xx);
  end;
                    2: begin
  str(sO/n:7:4,xx) ;
  outtextxy(500,320,'Distanta normalizata medie:'+xx);
```

```
  outtextxy(500,335,'Valoarea critica (95%): 2.45');
                              end;
end;
outtextxy(300,450,'Pentru continuare apasati orice tasta!');
car:^readkey;
restorecrtmode;
setgraphmode(getgraphmode) ;
if opty=2 then begin s3(s,sd,1);eli (2) end;
settextjustify(1,1) ;
assign(g,'para.dat');
reset(g) ;
readin(g,alfaO,betaO) ;
readln(g,alfa0,beta0,gamaO,deltaO) ;
beta0:=bet;
close(g) ;
axe;
outtextxy(40,40,'eh2 [mm]');
for i:=l to n do begin
 circle(30+round(0.1*t0[i]),400-round(5*(x2[i]-x2 [1])),2);
 setcolor (11) ;
 circle(30+round(0.7 *(tO[i]+15)),400-round(5*yc2[i] ) , 2) ;
 seteolor(15)
                              end;
x0:=30;y0:=400;
alfa:=alfa0;beta;=beta0 ;
gama:=gama0;delta:=delta0;
moveto(xO,yO);
case opty of 1: s2(s0,sd,0); 2: s4(s0,sd,0); end;
for i;=l to n do begin
 x0:-30+round(0.7*(i-1)*15) ;
 y0:-400-round(5*y2[i]) ;
lineto(x0,y0) ;
                              end;
strfalfa0:8:3,xx);str(beta0:ll:8,yy);
str(gama0:8:3,tt);str(deltaO:ll:8,uu);str(s0:8:4,zz);
outtextxy(320,430,'B='+xx+'  al='+yy+'  C='+tt+'  a2='+uu+'  S='+zz)
outtextxy (300,450,'Pentru continuare apasati orice tasta!');
car:=readkey;
setcolor (0);
for i:=l to n do begin
 circle(30+round(0.7*10 [i]),400-round(5*(x2[i]-x2[1])),2);
   circle(30+roundfO.7*(tO[i]+15)),400-round(5*yc2[i]),2)
                               end;
xO:=30;y0:=400;
alfa:=alfa0;beta:=beta0;
gama:=gama0;delta:=delta0 ;
moveto (xO, yO) ;
case opty of 1: s2(s0,sd,0); 2: s4(s0,sd,0); end;
for i:=1 to n do begin
 xO:-30+round(0.7*(i-1)*15) ;
 y0:=400-round(5*y2[i]);
 lineto(x0,y0) ;
                              end;
outtextxy(300,450,'Pentru continuare apasati orice tasta!');
setcolor(15);
k:=0;
repeat
 x0:=30;y0;=400;
 alfa:=alfa0+0.05*(1.0-2.O*random) ;
 beta:=beta0;
 gama:=gama0+0.05*(1.0-2.O*random) ;
 delta:=delta0+0.000005*(1.0-2.0*random) ;
 moveto(x0,y0) ;
```

```
 case opty of 1: s2(s,sd,0); 2; s4(s,sd,0); end;
  for i:=l to n do begin
    x0:=30+round(0.7*(i-1)*15);
y0:=400-round(5*y2[i]) ;
lineto(x0,y0) ;
                        end;
 for i: =1 to n do
circle(30+round(0.7*t0[i]),400-round(5 *(x2[i]-x2[1])),2) ;
 delay (20) ;
  setcolor (0) ;
  x0:=30;y0:=400;
  moveto(xO,yO);
  case opty of 1: s2(s,sd,0); 2: s4(s,sd,0); end;
   for i:=l to n do begin
     x0:=30+round(0.7*(i-1)*15) ;
  yO:=400-round(5*y2[i] ) ;
  lineto (x0,y0) ;
  end;
  setcolor(15);
   if s0>s then begin
     alfa0:=alfa;beta0:=beta;
  gamaO:=gama;delta 0:=delta;
  s0:=s;k:=l
                    end else k:=k+l;
   if k=10*(k div 10) then begin
     if k>0 then
     begin setcolor (0) ,-outtextxy (500, 320,xx+yy) ; setcolor (15) end;
  str(s:10:3,xx);str(k:5,yy);outtextxy(500,320,xx+yy)
  end;
  until k=nit;
 for i:=l to n do
   circle(30+round(0.7*t0[i]),400-round(5*(x2[i]-x2[1] ) ) ,2) ;
  x0:=30;y0:=400;
alfa:=alta0;beta:=beta0;
gama:=gama0;delta:=delta0;
moveto(xO,yO);
case opty of 1: s2(s0,sd,0); 2: s4(s0,sd,0); end;
for i:=l to n do begin
x0:=30+round(0.7*(i-l)*15) ;
y0:=400-round(5*y2[i]) ;
lineto(x0,y0) ;
end;
setcolor(0);outtextxy(500,320,xx+yy);setcolor (14) ;
strfalfaO:8:3,xx);str (betaO:11:8,yy) ;
str(gamaO:8:3,tt);str(delta0:ll:8,uu);str(s0:8:4,zz);
outtextxy(320,440,'B='+xx+' al='+yy+' C='+tt+' a2='+uu+' S='+zz);
setcolor (15) ;
case opty of 1: begin
str (s0/ (n-3) : 6: 4,xx) ;
str(n-3:3,yy) ;
outtextxy(500,320,'sy='+xx) ;
outtextxy(500,335,'Grade de libertate: '+yy);
str(sqrt(s0/(n-3)):6:3,xx) ;
outtextxy(500,350,'Abatere medie patratica: '+xx);
end;
2: begin
str(sO/n:1:4,xx) ;
outtextxy(500,320,'Distanta normalizata medie:'+xx);
outtextxy (500,335,'Valoarea critict (95%): 2.45');
end;
```

```
end;
rewrite(g) ;
writeln(g,alf:10:3,' ',bet:12:8) ;
writeln(g,alfa0:10:3,' ',betaO:12:8,' ',gamaO:10:3,' ',deltaO:12:8) ;
close(g) ;
outtextxy(300,450,'Pentru continuare apasati orice tasta!');
car:=readkey;
closegraph;
assign (f,'est.dat') ;
rewrite(f);
case opty of 1: writelnff,' t      hi    hie    ehl     h2    h2c    eh2')
                     2:   writeln(f,'   t                hi           hie           rol              h2          h2c          ro2')
end;
for i:=l to n do
 case opty of 1:
wnteln(f,t0 [i] : 5 : l,xl [i] : 8 :1, yl [i]+xl[l] : 7 :2 , yl [i]+xl [1]-xl [i] :6:2
                                ,x2 [i] :8:l,y2 [i]+x2 [1] : 7 : 2 , y2 [i]+x2 [ 1]-x2 [i] :6:2) ;
  2:
  writeln(f,t0[i] :5:l,xl [i]:8:1,yl[i]+xl[1]:7:2,rol[i]:6:3
                                ,x2[i]:8:l,y2[i]+x2[1]:7:2,ro2[i]:6:3);
    end;
   close(f)

 end.
```

```
program rec;
 uses CRT,graph;
 { Fisierul EGAVGA.BGI trebuie sa fie prezent in directorul/folderul de lucru(
const
 nf=32;
 model:array [1..2] of string= ( ' y (t+1) =cl. y (t)+sl. u (t) ' ,
 'y(t+l)=cl.y(t)+c2.y(t-l)+sl.u(t) ' ) ;
 para:array [1..2] of string= ( ' e ' '= [cl aal ] ' , ' e ' ' ='[cl c2 sl ] ' ) ;
  fata:array [l..nf] of string=(
 ' PROGRAM   PENTRU   ESTIMAREA   DE   PARAMETRI',
 '                          RECURSIVA',

' Programul prezent ilustreaza estimarea recursiva a parametrilor in doua',
'modele, unul de ordinul intai, altui de ordinul al doilea.',
' Datele experimentale din fisierul EXPE.DAT sunt recoltate efectiv de pe ',
'un sistem de doua vase cu scurgere libera, in cascada. Primui vas detasat',
'din context este (aproximativ) un sistem de ordinul unu, cu o intrare',
'(debitui de alimentare de la reteaua de apa curenta) si o iesire (nivelul',
'in vas). Ansamblul celor doua vase este (tot aproximativ) un sistem de',
'ordinul doi, de asemenea cu o intrare (aceeasi ca mai sus) si una sau doua',
' iesiri (nivelul in unul din vase sau ambele niveluri).',
' Sunt de estimat doi sau, respectiv trei parametri. Modelele discretizate',
'apar pe ecran la momentui potrivit. La fei vectorul parametrilor de esti-',
'mat e.',
' Relatia de recurenta este',
                                        e (t+l)=e (t)+K(t+l) .i(t+l) ',
' cu' ,
                          K(t+l)=P(t) .z(t+l)/(a(t+l)+z' ' (t+1) .P(t) .z(t+l)) ',
'in care ' ,
                                        a(t+1)=a0.a(t)+(l-a0) ',
' si cu ' ,
                                      i (t+l)=y(t+l)-z" (t+1) .e(t) ' ,
' Matricea P se calculeaza recursiv cu relatia',
'P(t+l)=[P(t)-P(t) .z(t+l) . (a(t+l)+z' ' (t+1) .P(t) .z (t+1) ).z"(t+l).P(t)]/a(t+l)'
' In cazurile tratate',
                                        z (t+l)=[-y(t) u(t) ] ',
'respectiv',
                                      z (t+l)=[-y (t) -y(t-l) u(t)]',
' Programul are in vedere o singura metoda, cea a celor mai mici patrate.',
','Pentru continuare apasati orice tasta!');
  drl:array [1..4] of pointtype=((x:70;y:150),(x:180;y:150)
```

```
                                                ,(x:180;y:210),(x:70;y:210)) ;
 dr2 : array [1..4] of pointtype" ( (x: 460; y: 170) , (x: 570,-y: 170)
                                                ,(x:570;y:230),(x:460;y:230)) ;
 pur:fillpatterntype=($FF,$FF,$FF,$FF,$FF,$FF,$FF,$FF) ;

 var
  f:text;
 gdriver,gmode,i,j,k,l,m,mx,my,n:integer;
 t0:array [1..60] of double;
 yl,yc,yest:array [1..60,1..2] of double;
 a,psi,tetal,tetaO:array [1..5] of double;
 b,pl,p0,pp:array [1..5,1..5] of double;
 alfa,la,la0,y:double;
 car:char;
 xx:string;

function ing(a:real;nd:integer):string;
 var
  xx,yy:string;
  i:integer;
 begin
 str(10.0*a,xx) ;
 yy:-";
 for i:=l to nd+6 do yy:=yy+' ';
 yy[l]:=xx[l] ;
 yy[2]:='.';
 yy[3] :=xx[2];
 if nd>l then for i:=2 to nd do yy[i+2]:=xx [i+2];
 yy[nd+3]:=xx(18] ;
 yy[nd+4]:=xx[19] ;
   yy[nd+5]:=xx[22];
   yy[nd+6]:-xx[23] ;
  ing:=' '+copy(yy,1/nd+6);
 end;

 procedure axe;
  var
   xx:string;
 begin
  for i:=0 to 1 do begin
  line(30,400+i,610,400+i) ;
 line(30-i,50,30-1,400) ;
 line(30-i,50,25-i,55) ;
 line (30-i,50,35-i,55) ;
 line(610,400+i,605,395+i);
 line(610,400+i,605,405+i)
 end;
 for i:=0 to 8 do begin
 line(30+70*1,400,30+70*1,397) ;
 str(100*i:3,xx) ;
 outtextxy(30+70*i,410,xx)
 end;
 outtextxy(615,410,'[s]') ;
 for i:=0 to 3 do begin
 line(30,400-100*1,33,400-100*1);
 str(20*i:2,xx) ;
 outtextxy (15,400-100*1,xx)
 end;
 end;

 procedure fatada;
```

```
 begin
   settextjustify(0,1) ;
   for i:=l to nf-1 do outtextxy(10,14*i,fata[i]);
   settextjustify(1,1);
   setcolor (14) ;
   outtextxy(320,14*nf,fata[nf]) ;
   setcolor(15)
  end;

 procedure recu(m:integer) ;
 begin
 setfillpattern(pur,15) ;
 fillpoly(4,drl);
 fillpoly(4,dr2) ;
 settextjustify(1,1) ;
 setcolor (0) ;
 line(75,150,75,210) ;
 line(465,170,465,230) ;
 line (75,180,175,180) ;
 line(465,200,565,200) ;
 case m of 2: outtextxy(85,155,'d');
           3: outtextxy(100,155,'cl,c2');
 end;
 outtextxy(475,175,'sl') ;
 setcolor (15) ;
 axe;
  str (m-1: 1, xx) ;
  outtextxy(40,40,'eh'+xx+' [mm]') ;
  outtextxy(320,425,'MODEL: '+model[m-1]) ;
  outtextxy(320,440,'VECTOROL PARAMETRILOR DE ESTIMAT: '+para[m-1]),
for i: =1 to n do
 circle(30+round(0.7*t0[i]),400-round(5*(yl[i,m-1]-yl[1,m-l])),2);
 for i:=l to 5 do tetal[i]:=0.0;
for i: =1 to 5 do
 for j:=l to 5 do begin
                  pi [i,j]:-0.0;if i=j then pl[i,j]:=1.0
                                end;
 for i:=l to m-1 do yest[i,m-l]:=0.0;
 la 0:=0.95;la:=0.95;
 settextjustify (0,1) ;
 circle(60,80,2) ;
 outtextxy(70,80, 'EXPERIMENTAL') ;
 setcolor(14);
 circle(60,95, 2) ;
 outtextxy(70,95,'ESTIMAT PAS CU PAS ' ) ;
outtextxy(70,475,
 'Treceti de la un moment la urmatorul apasand o tasta oarecare!');
 setcolor(13);
 circle (60,110,2) ;
 outtextxy(70,110,'CALCULAT CU MODELUL FINAL');
 setcolor(15) ;
 settextjustify(1,1) ;
 outtextxy(386+36*m,290,'MATRICEA P');
 line(390,300,390,310+10-m);
 line (390, 300, 3 93, 300) ;
 line(390,310+10*m,393,310+10*m) ;
 line(400+72*m,300,400+72*m,310+10*m);
 line(397+72*m,300,400+72*m,300);
 line(397+72*m,310+10*m,400+72*m,310+10*m) ;
 outtextxy(336,55+5*m,'e =');
 outtextxy(328,50+5*m,'"');
 line(400-44,50,400-44,60+10*m);
```

```
line(400+54,50,400+54,60+10*m);
line(356,50,359,50) ;
line(356,60+10*m,359,60+10*m) ;
line(451,50,454,50);
line(451,60+10*m,454,60+10*m) ;
for k:=m to n do begin
  for i;=l to m do for j:=l to m do pO[i,j]:^pl[i,]] ;
for i:=l to m do tetaO[i]:=tetal[i] ;
  for i:=l to m do begin
    for j:=l to m do outtextxy(350+72*i,300+10*j,ing(pO[i,j],3));
outtextxy(400,50+10*i,ing(tetaO[i],5)) ;
end;
 for i:=l to m-1 do putpixel(75+2*(k-m),180-round(20*teta0[i]),2*(i-1))
 putpixel(465+2*(k-m),200-round(300*teta0[m]),0) ;
car:=readkey;
setcolor(0) ;
  for i:=l to m do begin
    for j:=l to m do outtextxy(350+72*i,300+10*j,ing(pO[i,j],3));
outtextxy(400,50+10*i,ing(teta0[i],5));
end;
setcolor(15) ;
for i:=l to m-1 do psi[i]:=-yl[k-i.m-1]+yl[l,m-l] ;
psi[m]:=30.0;
y:=yl[k,m-l]-yl[l,m-l] ;
for i:=l to m do begin
a[i]:=0.0;
    for j:=l to m do a[i]:=a[i]+psi[i]*p0[j,i]
                          end;
la:=la0*la+l.0-laO;
alfa:=la;
for i =1 to m do alfa:=alfa+a [i]*psi[i] ;
alfa:=1.0/alfa;
for i =1 to m do for j:=1 to m do pp[i,j]:=alfa*psi[i]*psi[j] ;
for i ^1 to m do for j :=1 to m do b[i,j]:=0.0;
for i =1 to m do for j:=1 to m do
for 1:=1 to m do b[i,j]:=b[i,j]+p0[i,l]*pp[1,j] ;
for i =1 to m do for j :=1 to m do pp [i, j ] :=0.0;
for i =1 to m do for j:=1 to m do
for 1:=1 to m do pp [i, j ] :=pp [i, j ]+b [i, 1] *p0 [1, j ] ;
                          for i:=l to m do for j:=l to m do pi [i, j ] := (pO [i , j ]-pp [i , j ] )/la;
yest[k,m-l]:=0.0;
for i:=l to m do yest[k,m-1]:=yest[k,m-l]+psi[i]*teta0[i];
setcolor(14) ;
circle(30+round(0.7*t0[k]),400-round(5*yest[k,m-l]),2);
setcolor(15) ;
for i:=l to m do a[i]:=0.0;
for i:=l to m do for 1:=1 to m do a [i] :-a [i]+p0 [i, 1] *psi [1] ;
for i:=l to m do begin
a[i]:=alfa*(y-yest[k,m-l])*a[i];
tetal[i):=teta0[i]+a[i]
end;
end;
if m=2 then assign(f,'reel.dat') else assign(f,'rec2.dat');
rewrite(f) ;
for i:=l to m-1 do yc[i,m-l]:=yl[i,m-l]-yl[l,m-l] ;
setcolor(13) ;
  for k:=m to n do begin
    yc[k,m-l]:=tetal[m]*30.0;
for i:=1 to m-1 do
yc[k,m-l]:=yc[k,m-l]-tetal[i]*yc [k-i,m-l] ;
circle(30+round(0.7*t0[k]),400-round(5*yc[k,m-1]),2) ;
                          wnteln(f,yl [k, m-1]-yl [ 1, m-1] : 10 : 2 , yest [ k,m-l] : 12 : 3 , yc [k,m-l] : 12 :3)
                                                                                                    end;
close(f) ;
```

```
  for i:=1 to m do begin
    for j:=l to m do outtextxy(350+72*i,300+10*j,ing(pO[i,j],3));
outtextxy(400,50+10*1,ing(teta0 [i],5)) ;
                          end;
car:=readkey;
clear viewport;
setcolor(15)
end;

begin
 assign (f,'expe.dat') ;
 reset (f) ;
 readin(f,n) ;
 for i:=l to n do readin(f,tO[i],yl[i,1],yl[i,2]);
 close(f) ;
 detectgraph(gdriver,gmode) ;
 InitGraph(gdriver,gmode,'') ;
 mx:=getmaxx;
 my:=getmaxy;
 fatada;
 car:=readkey;
 clear viewport;
 recu(2) ;
 recu(3) ;
 closegraph
end.
```

LUCRAREA 13

IDENTIFICARE DINAMICĂ PENTRU CONDUCEREA UNUI SISTEM DE REGLARE A PRESIUNII

1. OBIECTIVELE LUCRĂRII

- Aplicarea tehnicilor de identificare dinamică a proceselor în scopul proiectării unui regulator pentru conducerea unui sistem de reglare a presiunii.
- Elaborarea unui program software de achiziţie de date şi identificare pentru un sistem de reglare a presiunii.

2. BREVIAR TEORETIC

2.1. Descrierea SRA-P

Sistemul de reglare automată ce urmează a fi studiat este destinat reglării presiunii aerului într-un recipient şi se află în laborator. Schema sistemului de reglare automată este prezentată în figura 13.1.

Traductorul de presiune PT sesizează continuu variaţia presiunii P_2 şi transmite regulatorului PC un semnal electric proporţional cu valoarea curentă a acestuia. Regulatorul compară valoarea semnalului I_r cu valoarea I_i asociată prescrierii P_i şi în cazul în care apare o abatere se emite o comandă I_c robinetului de reglare, după un algoritm de tip PI sau PID. Convertorul electro-pneumatic face posibilă compatibilitatea funcţionării regulatorului PC – electronic - cu robinetul de reglare RR-pneumatic.

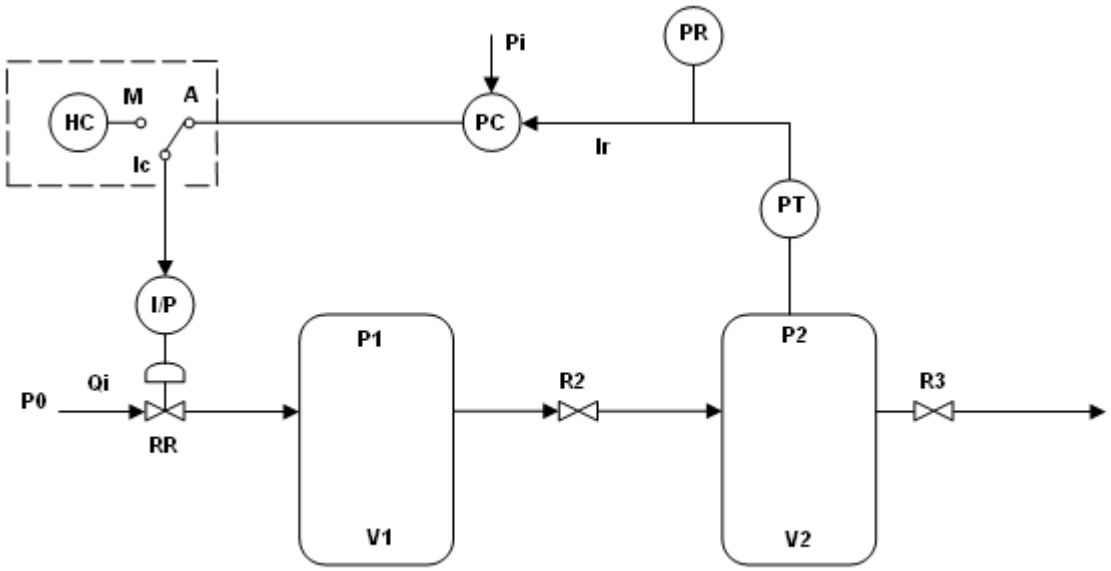


Fig. 13.1. Schema sistemului de reglare automată a presiunii.

Sistemul de reglare este dublat de un element de comandă manuală HC (conturul punctat). Comutatorul ce selectează cuplarea/decuplarea sistemului de reglare este amplasat pe elementul de comandă manuală HC.

2.2. Modelul matematic al procesului

În regim dinamic cele două vase sunt descrise de ecuaţiile:

$$V_1 \cdot \frac{d\rho_1}{dt} = Q_{1i} - Q_{1e}; \tag{13.1}$$

$$V_2 \cdot \frac{d\rho_2}{dt} = Q_{2i} - Q_{2e} \equiv Q_{1e} - Q_{2e}. \tag{13.2}$$

Curgerea prin robinetele R_2 şi R_3 fiind turbulentă, poate fi scrisă relaţia

$$Q_{1e} = K_V \cdot \sqrt{\rho_2 (P_1 - P_2)} \tag{13.3}$$

De asemenea, se cunoaşte

$$\rho = \frac{M}{RT} \cdot P, \tag{13.4}$$

Înlocuind relaţiile (13.3) şi (13.4) în (13.2), rezultă:

$$\frac{V_1 M}{RT}\cdot\frac{dP_1}{dt} = Q_{1i} - K_{V2}\cdot\sqrt{\frac{M}{RT}\cdot P_1(P_1-P_2)};\cdot \tag{13.5}$$

$$\frac{V_2 M}{RT}\cdot\frac{dP_2}{dt} = K_{V2}\cdot\sqrt{\frac{M}{RT}\cdot P_1(P_1-P_2)} - Q_{2e}. \tag{13.6}$$

care, liniarizate în jurul punctului $(P_{10};P_{20};Q_{1i0};Q_{2e0})$, conduc la:

$$\begin{aligned}
&\frac{V_1 M}{RT}\Delta\dot{P}_1 + K_{V_2}\sqrt{\frac{M}{RT}}\frac{2P_{10}-P_{20}}{2\sqrt{P_{10}^2-P_{10}P_{20}}}\Delta P_1 = \Delta Q_{1i} + K_{V_2}\sqrt{\frac{M}{RT}}\frac{P_{10}}{2\sqrt{P_{10}^2-P_{10}P_{20}}}\Delta P_2\\
&\frac{V_2 M}{RT}\Delta\dot{P}_2 + K_{V_2}\sqrt{\frac{M}{RT}}\frac{P_{10}}{2\sqrt{P_{10}^2-P_{10}P_{20}}}\Delta P_2 = K_{V_2}\sqrt{\frac{M}{RT}}\frac{2P_{10}-P_{20}}{2\sqrt{P_{10}^2-P_{10}P_{20}}}\Delta P_1 - \Delta Q_{2e};
\end{aligned} \tag{13.7}$$

respectiv:

$$\begin{aligned}
a_1\Delta\dot{P}_1 + \Delta P_1 &= b_{11}\Delta Q_{1i} + b_{12}\Delta P_2\\
a_2\Delta\dot{P}_2 + \Delta P_2 &= b_{21}\Delta P_1 + b_{22}\Delta Q_{2e}
\end{aligned} \tag{13.8}$$

unde:

$$\begin{aligned}
a_1 &= \frac{V_1 M}{RT}\sqrt{\frac{RT}{M}}\frac{2\sqrt{P_{10}^2-P_{10}P_{20}}}{K_{V_2}(2P_{10}-P_{20})}\\
a_2 &= \frac{V_2 M}{RT}\sqrt{\frac{RT}{M}}\frac{2\sqrt{P_{10}^2-P_{10}P_{20}}}{K_{V_2}P_{10}}\\
b_{11} &= \sqrt{\frac{RT}{M}}\frac{2\sqrt{P_{10}^2-P_{10}P_{20}}}{K_{V_2}(2P_{10}-P_{20})}\\
b_{22} &= \sqrt{\frac{M}{RT}}\frac{2\sqrt{P_{10}^2-P_{10}P_{20}}}{K_{V_2}P_{10}}\\
b_{12} &= \frac{P_{10}}{2P_{10}-P_{20}};\\
b_{21} &= \frac{2P_{10}-P_{20}}{P_{10}};
\end{aligned} \tag{13.9}$$

Valori practice pentru constantele de timp şi coeficienţii de amplificare asociate instalaţiei din laborator sunt considerate a fi următoarele:

$$V_1 = V_2 = V = \frac{\pi D^2}{4}\cdot H = \frac{\pi\cdot 0{,}3^2}{4}\cdot 0{,}5 = 0{,}035\ \ m^3;$$

$P_1 = 2$ bar $= 2\times10^5 N/m^2$; $P_2 = 1$ bar $= 10^5 N/m^2$;

$$\rho_1 = \frac{M}{RT}P_1 = \frac{29\times2\times10^5}{8315\times293} = 2{,}38 \quad Kg/m^3;$$

$$Q = \frac{\pi d^2}{4}\times\rho_1 = \frac{\pi\times0{,}006^2}{4}\times15\times2{,}38 = 0{,}001 \quad Kg/s;$$

$$K_{V_2} = \frac{Q}{\sqrt{\rho_1(P_1-P_2)}} = \frac{0{,}001}{\sqrt{2{,}38(2-1)\times10^5}} = 2\times10^{-6} \quad m^2;$$

$$a_1 = \frac{0{,}035\times29}{8315\times293}\sqrt{\frac{8315\times293}{29}}\times\frac{2\sqrt{2\times10^5(2-1)\times10^5}}{2\times10^6\cdot(4\times10^5-10^5)} \approx 60 \quad s;$$

$$a_2 = \frac{0{,}035\times29}{8315\times293}\sqrt{\frac{8315\times293}{29}}\times\frac{2\sqrt{2\times10^5(2-1)\times10^5}}{2\times10^6\cdot2\times10^5} \approx 85 \quad s;$$

$$b_{11} = \sqrt{\frac{8315\times293}{29}}\,\frac{2\sqrt{2\times10^5(2-1)\times10^5}}{2\times10^6\cdot(4-1)\cdot10^5} = 1{,}36\times10^8 \quad \frac{1}{ms};$$

$$b_{12} = \frac{P_{10}}{2P_{10}-P_{20}} = \frac{2}{2\times2-1} = 0{,}67;$$

$$b_{21} = \frac{2P_{10}-P_{20}P_{10}}{P_{10}} = \frac{2\times2-1}{2} = 1{,}5;$$

$$b_{22} = \sqrt{\frac{8315\times293}{29}}\,\frac{2\sqrt{2\times10^5(2-1)\times10^5}}{2\times10^6\cdot2\times10^5} = 2{,}05\times10^8 \quad \frac{1}{ms}.$$

Este evident că valorile constantelor $a_1, a_2, b_{11}, b_{12}, b_{21}, b_{22}$ pot fi modificate, printre altele, cu ajutorul robinetului R_2.

Având în vedere că cele două vase sunt în interacţiune, fapt evidenţiat şi de modelul matematic, schema bloc a procesului, figura 13.2, conţine un element cu reacţie.

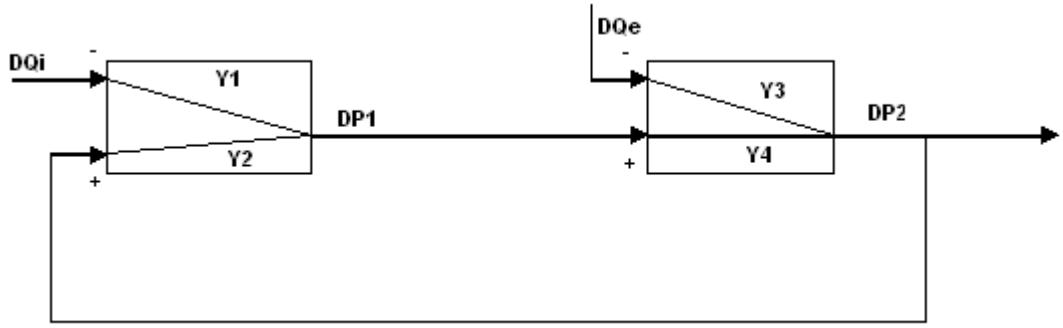


Fig. 13.2. Schema bloc a procesului de acumulare a gazului în cele două vase cu interacţiune

Sistemul de ecuaţii diferenţiale care reprezintă modelul şi schema bloc din figura 13.2 permit evidenţierea următoarelor funcţii de transfer:

$$\begin{aligned} Y_1(s) &= \frac{\Delta P_1(s)}{\Delta Q_1(s)} = \frac{b_{11}}{a_1 s + 1}, \\ Y_2(s) &= \frac{\Delta P_1(s)}{\Delta P_2(s)} = \frac{b_{12}}{a_1 s + 1}, \\ Y_3(s) &= \frac{\Delta P_2(s)}{\Delta Q_e(s)} = -\frac{b_{22}}{a_2 s + 1}, \\ Y_4(s) &= \frac{\Delta P_2(s)}{\Delta P_1(1)} = \frac{b_{21}}{a_2 s + 1}. \end{aligned} \qquad (13.10)$$

În ansamblu, procesul reprezintă un sistem caracterizat prin funcţii de transfer ce fac legătura între mărimea de execuţie, mărimea perturbatoare şi mărimea de ieşire.

Pentru procesul de acumulare a gazului din cadrul SRA-P schema bloc este adusă la forma din figura 13.3. Aplicând algebra funcţiilor de transfer pentru sisteme cu reacţie se poate scrie:

$$\Delta P_2(s) = Y_1 Y_4 \cdot \Delta Q_1(s) + Y_2 Y_4 \cdot \Delta P_2(s) - Y_3 \cdot \Delta Q_e(s) \qquad (13.11)$$

ceea ce conduce la

$$\Delta P_2(s) = \frac{Y_1 Y_4}{1 - Y_2 Y_4} \Delta Q_1(s) - \frac{Y_3}{1 - Y_2 Y_4} \Delta Q_e(s) \qquad (13.12)$$

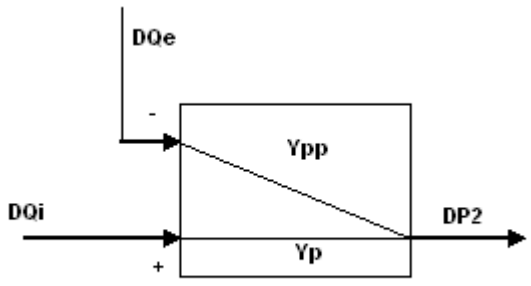


Fig. 13.3. Schema bloc simplificată a procesului

Din relaţia de mai sus se deduc funcţiile de transfer asociate procesului, de forma:

$$Y_P(s)=\frac{\Delta P_2(s)}{\Delta Q_1(s)}=\frac{Y_1Y_4}{1-Y_2Y_4}$$
$$Y_{PP}(s)=\frac{\Delta P_2(s)}{\Delta Q_e(s)}=-\frac{Y_3}{1-Y_2Y_4} \qquad (13.13)$$

2.3. Identificarea dinamică în contextul proiectării unui sistem de reglare

Identificarea este operaţia de determinare a caracteristicilor dinamice ale procesului (sistemului), a cărui cunoaştere este necesară pentru proiectarea şi implementarea unui sistem performant de reglare.

Figura 13.4 rezumă principiile generale de proiectare si calcul ale unui regulator.

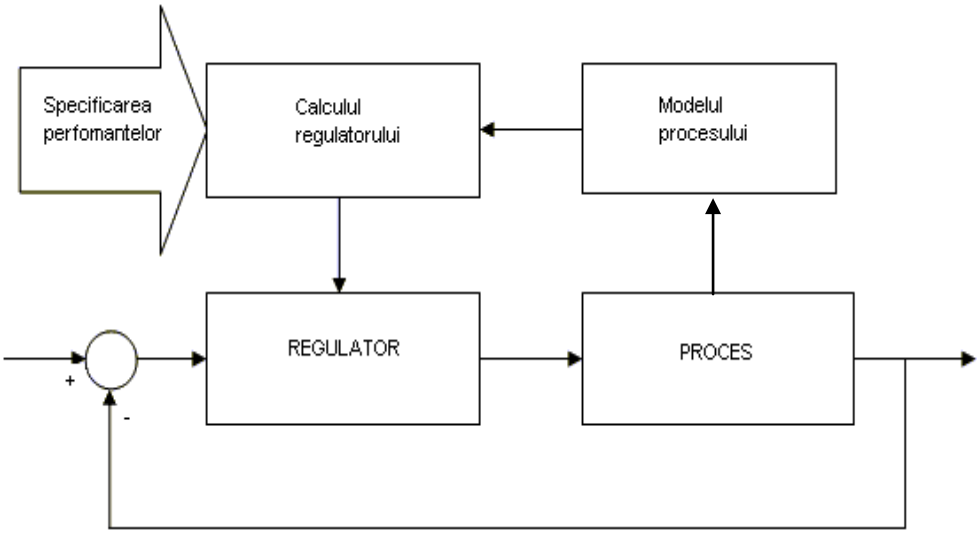


Fig. 13.4. Principiul proiectării şi calculului unui regulator.

Pentru a proiecta corect un regulator sunt necesare:

- specificarea performanţelor dorite pentru bucla de comandă/reglare;

- cunoaşterea modelului dinamic al procesului (numit model de comandă) ce descrie relaţia între variaţiile comenzii şi variaţiile ieşirii;

- disponibilitatea unei metode adecvate de calcul al regulatorului compatibil cu performanţele specificate şi caracteristicile modelului procesului.

Noţiunea de *model matematic* al unui sistem sau fenomen este un concept fundamental. În general, există diferite tipuri de modele, fiecare model fiind destinat unei aplicaţii specifice.

De exemplu, modelele de cunoaştere (bazate pe legile fizice, chimice, etc.) permit o descriere destul de completă a sistemelor şi sunt utilizate pentru simularea şi modelarea proceselor. Aceste modele sunt, în general, extrem de complexe şi rareori direct utilizabile în automatică. Modelele dinamice de comandă, ce dau relaţia între variaţiile intrărilor unui sistem şi variaţiile ieşirii, sunt tipuri de modele necesare pentru proiectarea şi ajustarea sistemelor de comandă/reglare. Deşi indicaţii asupra structurii acestor modele de comandă se pot obţine pornind de la structura modelului de cunoaştere, în general, este foarte dificil să se determine valorile parametrilor semnificativi pornind de la aceste modele.

De aceea, în marea majoritate a situaţiilor practice, este pusă in aplicare o metodologie de identificarea directă a acestor metode dinamice (de comandă). De notat că modelele de comandă sunt de două tipuri:

- modele neparametrice (răspuns în frecvenţă, răspuns la treaptă).
- modele parametrice (funcţie de transfer, ecuaţie diferenţială sau cu diferenţe).

Metoda de identificare clasică utilizată pentru obţinerea modelelor parametrice pornind de la modele neparametrice de tip “răspuns la treaptă” este prezentată în figura 13.5.

Această metodă a fost utilizată iniţial pentru a obţine modele parametrice continue, apoi a fost extinsă pentru identificarea modelelor discrete. Pornind de la forma răspunsului procesului la

treaptă, se alege un tip de model şi se determină grafic parametrii modelului. Cunoscând frecvenţa de eşantionare, se poate obţine cu ajutorul tabelelor modelul discret corespunzător.

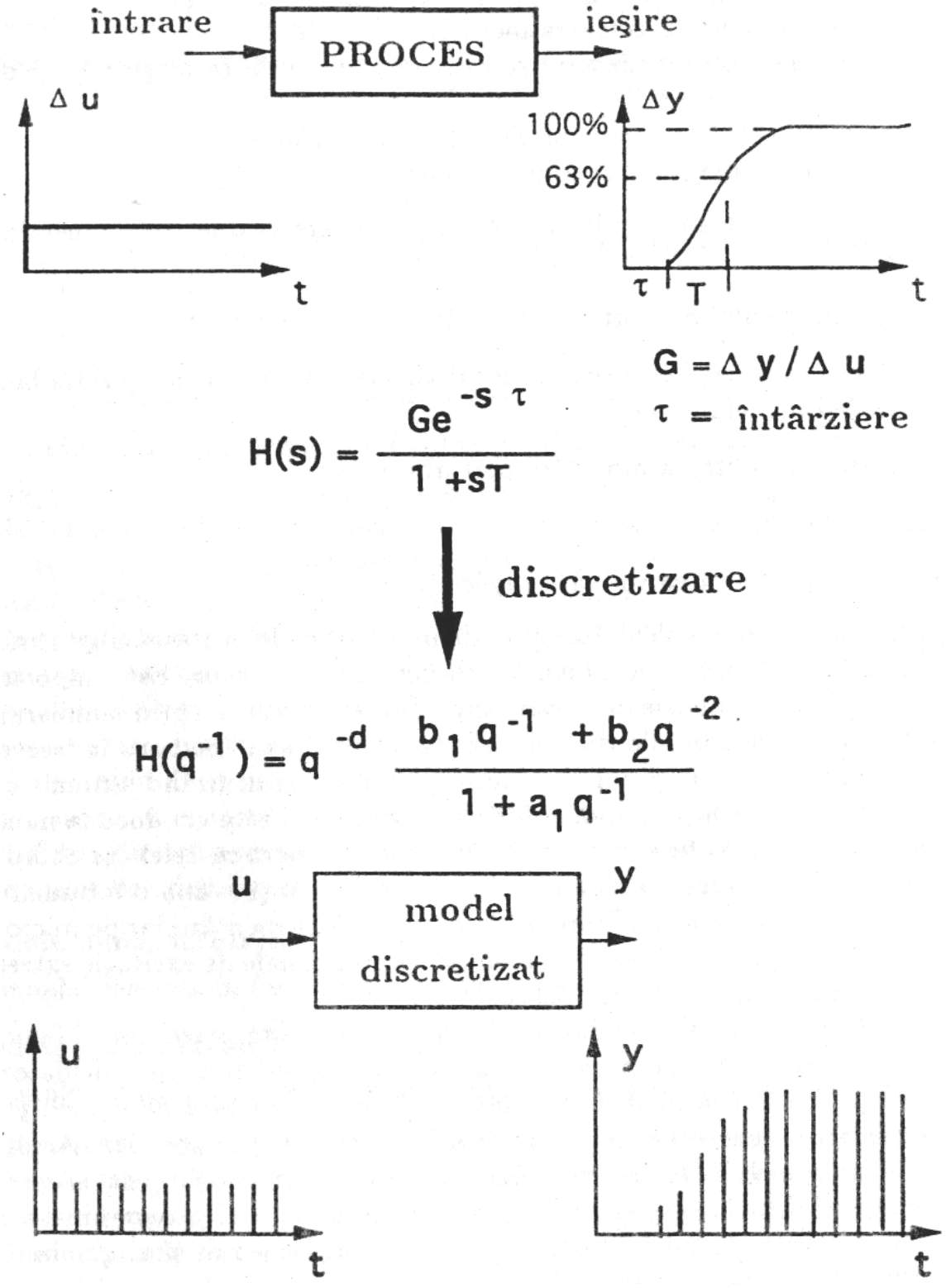


Fig. 13.5. Metoda clasică de identificare.

Principiul estimării parametrilor modelelor discrete este prezentat în figura 13.6.

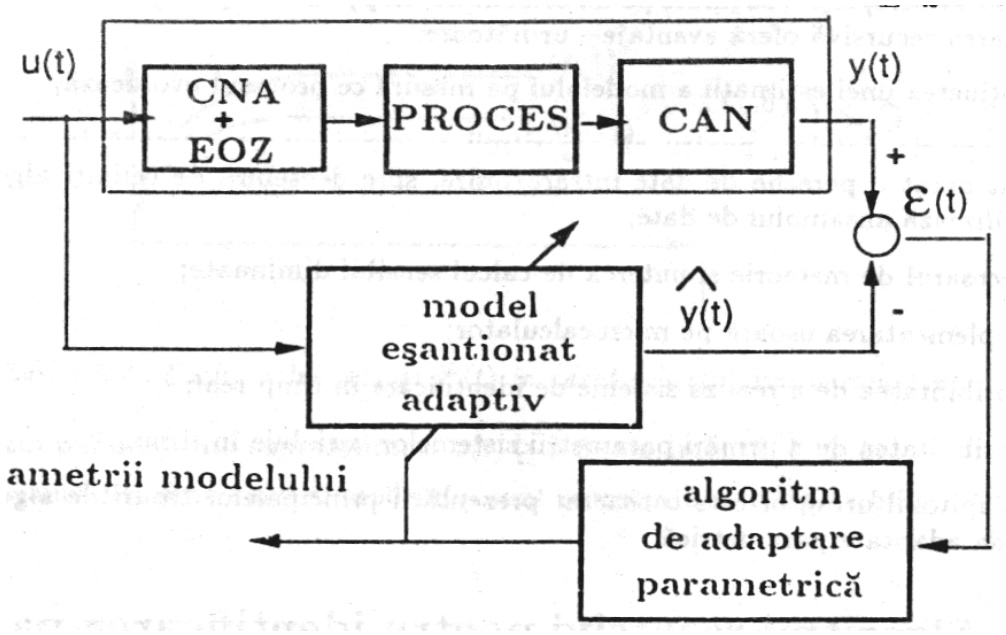


Fig. 13.6. Principiul estimării parametrilor unui model

Un model discret cu parametri ajustabili este implementat pe calculator. Eroarea între ieşirea procesului la momentul t, $y(t)$, şi ieşirea predictată de model, $\hat{y}(t)$, numită *eroare de predicţie*, este utilizată de algoritmul de aşteptare parametrică. Acesta va modifica parametrii modelului la fiecare moment de eşantionare, astfel încât să se minimizeze această eroare. Intrarea este, în general, o secvenţă pseudo – aleatoare binară de un nivel foarte slab, generată de calculator (succesiune de impulsuri dreptunghiulare cu durată aleator variabilă). Odată modelul obţinut, o validare obiectivă poate fi făcută prin teste statistice asupra erorii de predicţie $\varepsilon(t)$ şi ieşirii predictate $\hat{y}(t)$. Testul de validare permite pentru un proces dat să se aleagă cel mai bun model, respectiv cea mai bună structură şi cel mai bun algoritm pentru estimarea parametrilor.

Calculând şi reprezentând grafic răspunsul la o treaptă şi răspunsul în frecvenţă al modelului identificat, se poate determina modelul continuu (răspuns la o treaptă sau răspuns în frecvenţă).

3. MODUL DE LUCRU

- Echipamentul de conducere este compus din calculator şi interfaţă AX5411.
- Se utilizează interfaţa AX5411 multifuncţională, cu următoarele caracteristici:

Subsistemul intrărilor analogice

Număr de intrări	16 simple (AI0 – AI15)
Rezoluţie	12 bit
Frecvenţa de achiziţie	60 kHZ max.
Timpul de conversie	A/D 15 μs max
Timpul de achiziţie	5μs max/canal
Domenii de intrare	± 10 V, ± 5V, ± 2,5V, ±1.25 V ± 0,625V, ±0,3125V selectabile software
Impedanţa de ieşire	> 10 MΩ, 50 pF
Neliniaritate	± 1 LSB
Eroare inerentă	± 1 LSB

Subsistemul ieşirilor analogice

Număr de ieşiri	2 (DA0, DA1)
Rezoluţie	12 bit
Frecvenţa	33 kHZ max.
Domenii de ieşire	0-5V, 0-10V selectabile hard
Curent de ieşire	5 mA max

Subsistemul ieşirilor/intrărilor numerice

Intrări numerice	24 (disponibile 8 DI0-DI7)
Ieşiri numerice	24 (disponibile 8 DO0-DO7)
Nivele intrare/iesire	compatibile TTL
Conector intrare/ieşire	50 pini

Caracteristici de interfaţă

Magistrala	compatibilă IBM PC AT
Biţi adresă utilizaţi	A9- A0
Adresă de bază (port)	300 hexa
Locaţii necesare	16 (octeţi)
Nivele de întrerupere	2,3,4,5,6,7 (controlabile soft)
Sursa de întreruperi	FINISH bit conversie A/D
Opţiuni DMA	DMA1 sau DMA3 selectabile hard

În figura 13.7 este prezentată asignarea pinilor în conectorul interfeţei AX5411. Adresa portului de bază pentru interfaţă este 0x300. Faţă de aceasta, interfaţa utilizează 16 adrese consecutive care constituie spaţiul de I/O (R=read, W=write) conform figurii 13.8.

Nume	I/O	Funcţie
AIx	Intrare	Canal x intrări analogice
DAx	Ieşire	Canal x ieşiri analogice
DIx	Intrare	Canal x intrări numerice
DOx	Ieşire	Canal x ieşiri numerice
+12V	Sursă	
-12V	Sursă	
AGND	Masă	Analogică
DGND	Masă	Numerică
+5VP	Sursă	PC +5V
+12VP	Sursă	PC +12V
-12VP	Sursă	PC -12V
EXTRG	Intrare	Trigger extern

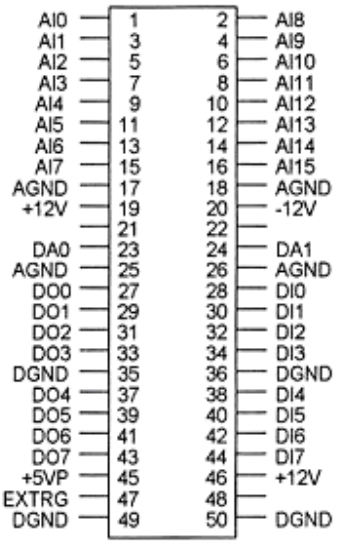


Fig. 13.7. Asignarea pinilor conectorului interfeţei AX5411

Locaţie	Funcţie	Tip
Adr. baza	A/D Low byte	R– stop conversie
+ 0	Start A/D	W –citeşte partea cea mai puţin semnificativă
+1	A/S High byte	R – 8 biţi cei mai semnificativi citiţi în cuante
	Gain Control	W– depind de temp. pe care îl folosesc
+ 2	Mux. scan control	R/W –canalul fixat
+ 3	Digital In	R– intrări
	Digital Out	W – valoarea de la ieşire
+ 4	D/A 0 Output Low byte	W –real în cuante a
+ 5	D/A Output Hight byte	W – sens pe care o generează
+ 6	D/A 1 Output Low byte	W
+ 7	D/A 1 Output High byte	W
+ 8	AX5411 Status	R
	Clear interrup	W
+ 9	AX5411 Control	R/W
+ 10	Digital Input Low byte	R
	Digital Output High byte	W
+ 11	Digital Input High byte	R
	Digital Output High byte	W
+ 12	8253 Counter 0	R/W
+ 13	8253 Counter 1	R/W
+ 14	8253 Counter 2	R/W
+ 15	8253 Counter Control	W

Fig.13.8. Spaţiul I/O al interfeţei AX5411

- Se studiază aplicaţia de achiziţie şi identificare a parametrilor modelului matematic, care este formată din trei secţiuni principale, descrise în continuare.

Programul de achiziţie permite generarea unor tensiuni către robinet şi achiziţionarea informaţiei primite de la traductorul de presiune (fig.13.9).

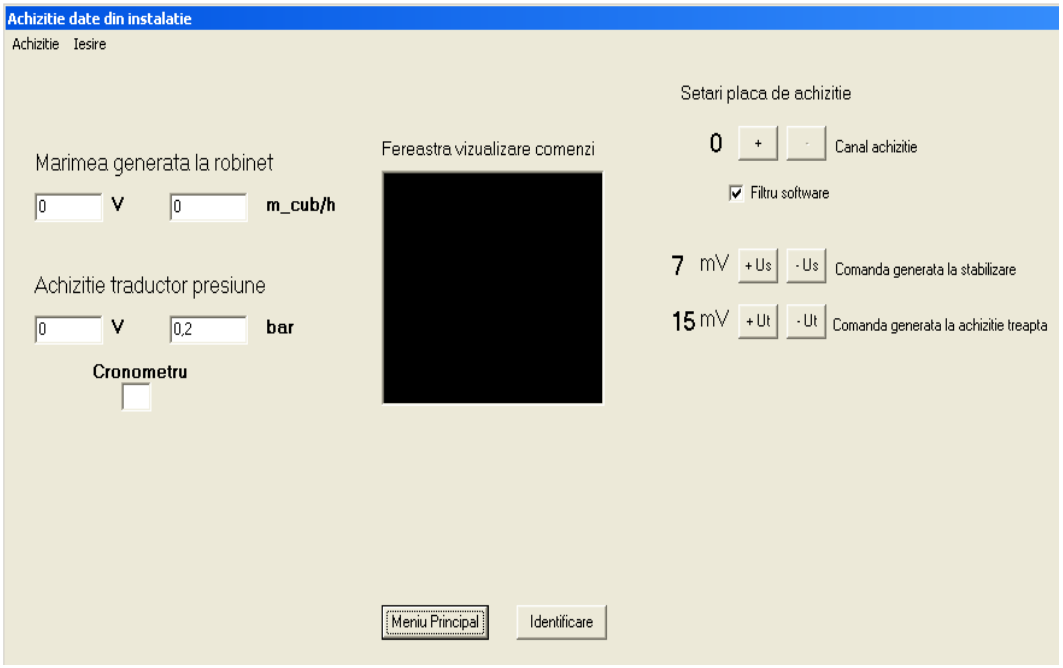


Fig. 13.9. Partea de achiziţie a aplicaţiei

Programul permite vizualizarea continuă a parametrilor instalaţiei şi a comenzii date către robinet prin indicatoarele aflate pe formă (fig.13.10).

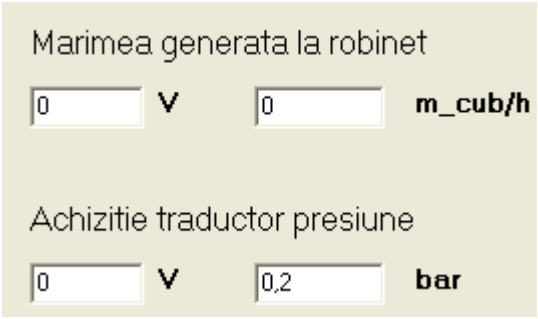


Fig. 13.10. Fereastra vizualizare date din proces

În modul de bază se face achiziţia de pe canalul 0, comanda de stabilizare este de 7 mA, comanda pentru treapta aplicată procesului este de 15 mA şi filtrul software este activ. Aceşti parametri pot fi schimbaţi utilizând butoanele din partea stângă a ferestrei, prezentate în figura 13.11.

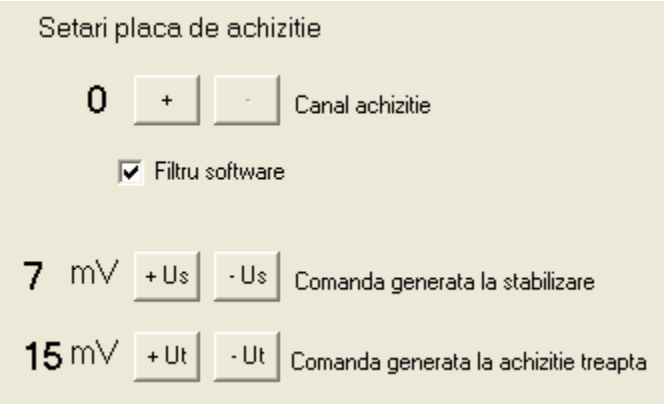


Fig. 13.11.Setări placa de achiziţie

Fiecare modificare este adusă la cunoştinţa utilizatorului prin intermediul ferestrei de vizualizare comenzi. În cazul când fereastra este plină, pentru a evita pierderea unei informaţii, se realizează întâi o ştergere a comenzilor anterioare şi apoi se afişează ultima comandă executată de către utilizator.

În figura 13.12 sunt prezentate unele comenzi executate urmând procedura descrisă anterior.

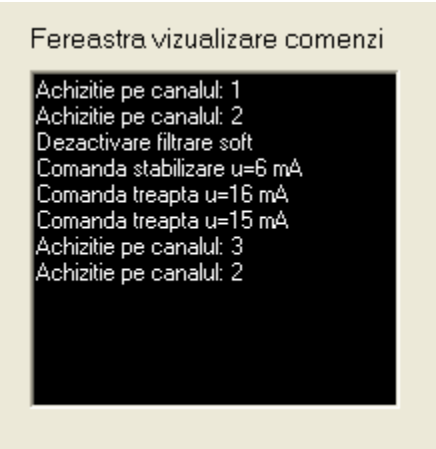


Fig.13.12. Fereastră vizualizare comenzi

Aplicaţia permite achiziţia datelor generând fie un semnal treaptă, fie unul rampă. Acest lucru se realizează utilizând meniul prezent în partea de sus a ferestrei, conform prezentării din figura 13.13..

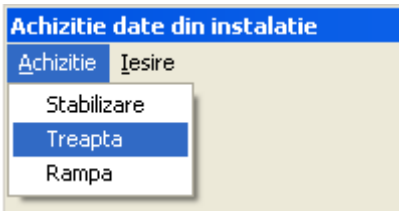


Fig. 13.13. Meniu generare semnale

Datele achiziţionate vor fi salvate în fişierele de tip text: „adi_tp.txt”, „adi_rp.txt”, direct în rădăcina disk-ului C.
Din forma de achiziţie se poate face o reîntoarcere în forma principală sau, după achiziţionarea datelor, trecere în secţiunea de identificare.

Programul de identificare are o fereastră asociată prezentată în figura 13.14.

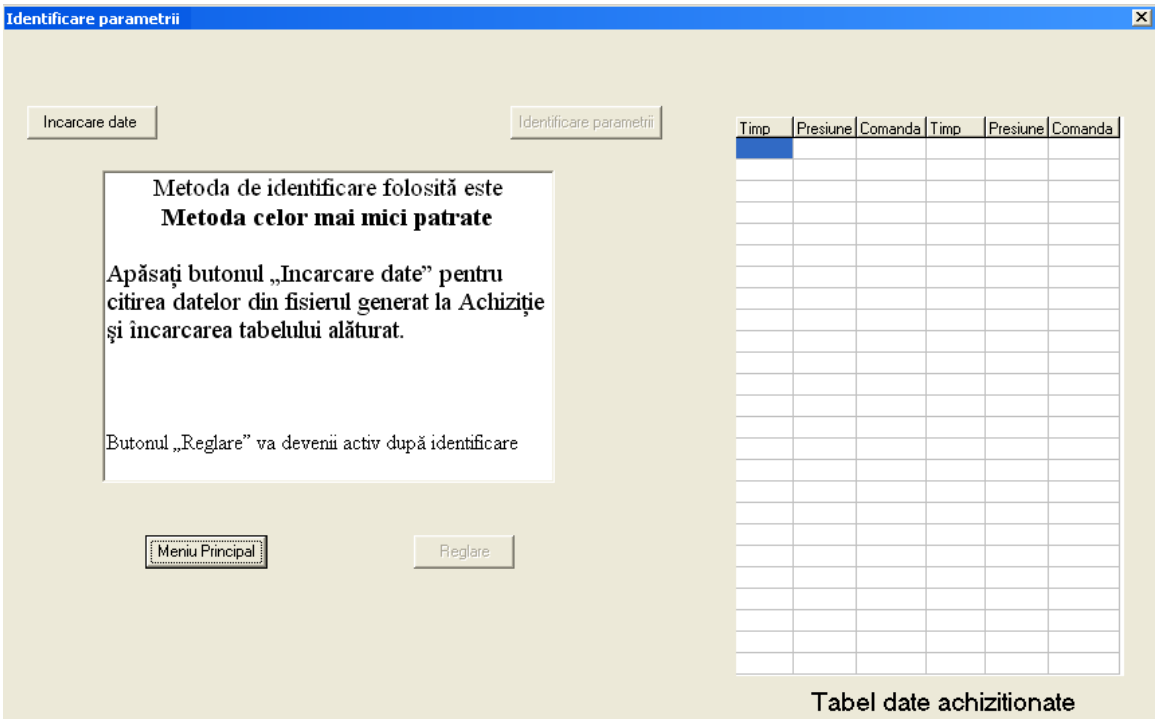


Fig.13.14. Fereastra asociată procedurii de identificare

Tabelul din figură va prelua datele achiziţionate din proces după apăsarea butonului „Incărcare date”. Un exemplu, utilizat la aplicarea procedurii de identificare pentru estimarea parametrilor modelului, este prezentat în figura 13.15. Butonul de identificare rămâne inactiv până când sunt preluate datele din fişierul text realizat în partea de achiziţie.

Timp	Presiune	Comanda	Timp	Presiune	Comanda
0	1.689	7.20	375	2.259	13.6
15	1.693	13.6	390	2.274	13.6
30	1.761	13.6	405	2.293	13.6
45	1.788	13.6	420	2.312	13.6
60	1.811	13.6	435	2.331	13.6
75	1.838	13.6	450	2.350	13.6
90	1.861	13.6	465	2.366	13.6
105	1.884	13.6	480	2.385	13.6
120	1.907	13.6	495	2.404	13.6
135	1.930	13.6	510	2.419	13.6
150	1.953	13.6	525	2.438	13.6
165	1.972	13.6	540	2.457	13.6
180	1.995	13.6	555	2.477	13.6
195	2.018	13.6	570	2.496	13.6
210	2.037	13.6	585	2.515	13.6
225	2.060	13.6	600	2.534	13.6
240	2.079	13.6	615	2.553	13.6
255	2.102	13.6	630	2.572	13.6
270	2.121	13.6	645	2.591	13.6
285	2.140	13.6	660	2.610	13.6
300	2.159	13.6	675	2.630	13.6
315	2.178	13.6	690	2.649	13.6
330	2.201	13.6	705	2.664	13.6
345	2.220	13.6	720	2.683	13.6
360	2.239	13.6	735	2.698	13.6

Fig.13.15. Tabelul cu datele achiziţionate

În urma derulării procedurii de identificare, pe ecran apare un indicator de tip ProgressBar care arată procentul realizat din identificare (fig.13.16), fereastra care prezintă modelul matematic actualizat în urma identificării (fig. 13.17) şi o aşa-numită fereastră de utilizator, care specifică separat valoarea parametrilor estimaţi în urma aplicării procedurii de identificare (fig.13.18).

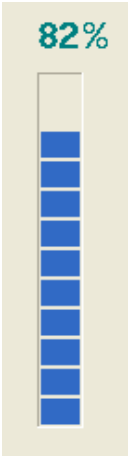


Fig. 13.16. Indicator proces identificare

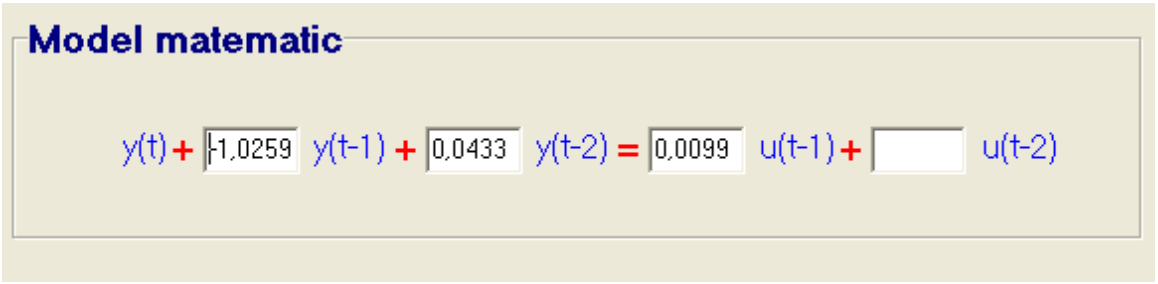


Fig. 13.17. Fereastra model matematic

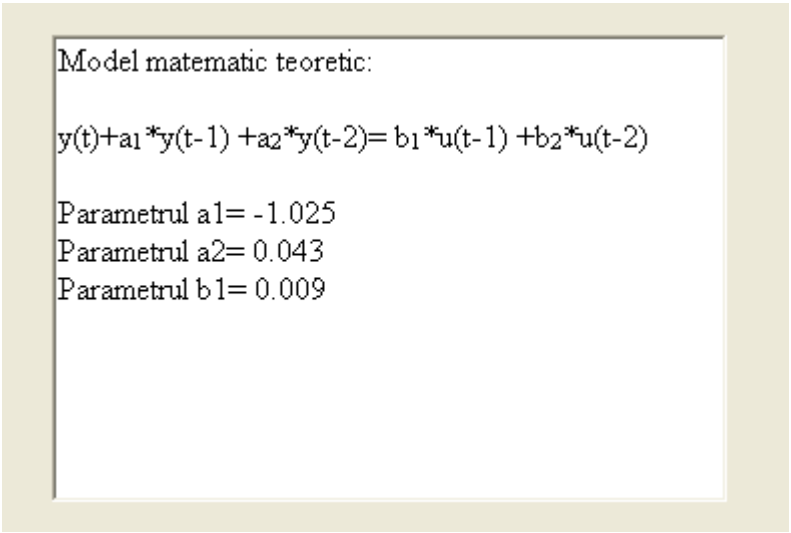


Fig. 13.18.Fereastra informare utilizator

După încheierea procedurii de identificare se poate trece la ultima parte a aplicaţiei: reglarea predictivă a sistemului, acţionând butonul „Reglare”.