Radu-Mihai Coliban Mihai Ivanovici

Prelucrarea semnalelor

Îndrumar de laborator

Editura Universității Transilvania din Braşov
2018

EDITURA UNIVERSITĂȚII TRANSILVANIA DIN BRAȘOV

Adresa: 500091 Braşov,
B-dul Iuliu Maniu 41A
Tel: 0268 476050
Fax: 0268 476051
E-mail: editura@unitbv.ro

Editură acreditată de CNCSIS
Adresa nr.1615 din 29 mai 2002

Ilustrație copertă: Alexandra Liana Stănescu
Copertă: Mihai Ivanovici

Descrierea CIP a Bibliotecii Naţionale a României
COLIBAN, RADU-MIHAI
Prelucrarea semnalelor : îndrumar de laborator / Radu-Mihai Coliban, Mihai Ivanovici. - Braşov : Editura Universităţii "Transilvania", 2018
Conţine bibliografie
ISBN 978-606-19-1039-7

I. Ivanovici, Mihai

004

Cuprins

Cuvânt înainte **1**

1 Introducere în Matlab **3**
1.1 Comenzi, operatori și variabile 3
1.2 Fișiere script și funcții 9
1.3 Structuri de control 10
1.4 Grafice 10
1.5 Generarea semnalelor periodice 11
1.6 Generarea unor semnale aleatoare 12
1.7 Exerciții 13

2 Serii Fourier **15**
2.1 Serii Fourier trigonometrice 15
2.2 Desfășurarea lucrării 17
2.3 Exercițiu 19

3 Transformata Fourier **21**
3.1 Relațiile de transformare 21
3.2 Desfășurarea lucrării 22
3.3 Exerciții 23

4 Eșantionarea semnalelor **25**
4.1 Teorema eșantionării 25
4.2 Desfăşurarea lucrării 29
4.3 Exerciții 31

5 Sisteme de timp discret **33**
5.1 Acumulatorul 33
5.2 Derivatorul de ordinul unu 34
5.3 Filtrul de mediere 36
5.4 Interpolatorul liniar 36

6 Filtrul trece-jos ideal **39**
6.1 Sisteme liniare invariante în timp 39
6.2 Filtrul trece-jos ideal 40
6.2.1 Filtrarea trece-jos ideală în domeniul frecvență 41
6.2.2 Filtrarea trece-jos ideală în domeniul timp 42
6.3 Desfășurarea lucrării 44
6.3.1 Filtrarea trece-jos în domeniul frecvență 45
6.3.2 Filtrarea trece-jos în domeniul timp 46
6.4 Exerciții 49

7 Modelarea semnalelor ca procese aleatoare **51**
7.1 Caracterizarea unui proces aleator 51
7.1.1 Momente statistice 52
7.2 Generarea proceselor aleatoare 52
7.3 Estimarea funcției de repartiție și a densității de probabilitate 53
7.4 Exerciții 55

8 Cuantizarea semnalelor **57**
8.1 Funcția de cuantizare 57
8.2 Cuantizarea uniformă 58
8.3 Cuantizarea optimală Lloyd–Max 59
8.4 Desfășurarea lucrării 61
8.4.1 Cuantizarea uniformă 61
8.4.2 Cuantizarea optimală Lloyd–Max 63
8.5 Exerciții 66

9 Corelația semnalelor **69**
9.1 Funcţia de autocorelaţie 69
9.1.1 Proprietăţile funcţiei de autocorelaţie 69
9.2 Funcţia de intercorelaţie 70
9.3 Densitatea spectrală de putere 71
9.4 Teorema Wiener-Hincin 71
9.5 Diagrama în spaţiul stărilor 72
9.6 Desfăşurarea lucrării 73
9.7 Exerciții 74

10 Filtrul adaptat la semnal **75**
10.1 Proiectarea filtrului adaptat la semnal 75
10.2 Desfăşurarea lucrării 76
10.3 Exerciții 78

11 Dreapta de regresie și analiza componentelor principale 81
11.1 Dreapta de regresie 81
11.1.1 Coeficientul de corelaţie 82
11.2 Analiza componentelor principale 83
11.3 Desfăşurarea lucrării 84
11.3.1 Componentele principale ale unei imagini color 86
11.4 Exerciții 88

12 Filtrarea inversă 91
12.1 Modelul de degradare 91
12.1.1 Modelul de degradare liniară 91
12.2 Filtrul invers de restaurare 95
12.2.1 Restaurarea prin filtrare inversă, în lipsa zgomotului . 95
12.2.2 Restaurarea prin filtrare inversă, în prezența zgomotului 97
12.3 Filtrul invers cu constrângeri 99
12.3.1 Restaurarea prin filtrare inversă cu constrângeri 100
12.4 Filtrul Wiener 101
12.5 Exerciții 103

Bibliografie selectivă 105

Cuvânt înainte

Prezentul îndrumar se adresează în principal studenților de la specializările Calculatoare, Tehnologia Informației, respectiv Electronică, Telecomunicații și Tehnologii Informaționale, toate programe de studii de licență la Facultatea de Inginerie Electrică și Știința Calculatoarelor din cadrul Universității Transilvania din Brașov. De asemenea, îndrumarul poate fi utilizat de specialiști din domeniul ingineriei sau informaticii interesați de aspectele practice, dar și teoretice ale procesării semnalelor.

Dorim să mulțumim Mariei Marincaș, studentă în anul 4 la specializarea Calculatoare din cadrul facultății mai sus menționate, pentru observațiile utile referitoare la Lucrarea 11, precum și Alexandrei Liana Stănescu pentru ilustrația de pe copertă. De asemenea, mulțumim tuturor studenților care, prin întrebările și observațiile lor, au contribuit la îmbunătățirea conținutului acestui îndrumar.

Autorii

Lucrarea 1

Introducere în Matlab

Scopul acestei prime lucrări îl reprezintă familiarizarea studenților cu mediile de dezvoltare Matlab și Octave și utilizarea acestora pentru generarea și vizualizarea unor semnale. Matlab reprezintă un limbaj de programare dedicat calculului numeric, precum și un mediu de dezvoltare ce oferă, pe lângă interpretorul Matlab, o serie de biblioteci și aplicații utile în diverse domenii, inclusiv procesarea semnalelor. Mediul de lucru Octave reprezintă o alternativă *open-source* la Matlab. Deși nu este 100% compatibil cu limbajul Matlab, Octave poate fi utilizat fără probleme în realizarea lucrărilor de laborator prezentate în acest îndrumar.

1.1 Comenzi, operatori și variabile

Mediul Matlab dispune de o interfață grafică ce conține câteva ferestre. Fereastra principală este *Command Window*, care oferă o linie de comandă pentru execuția instrucțiunilor și a scripturilor Matlab. Instrucțiunile se introduc în linia de comandă după prompter:

```
>>
```

execuția acestora realizându-se după apăsarea tastei *Enter*.

Sesiunea de lucru se desfășoară în directorul de lucru curent, în care sunt stocate toate fișierele utilizate în cadrul sesiunii. Calea către directorul curent poate fi afișată utilizând comanda:

```
>> pwd

ans=

C:\Users\user\Documents\MATLAB
```

Schimbarea directorului de lucru se poate face fie prin interfața grafică, din fereastra principală, fie prin linia de comandă, astfel:

```
>> cd D:\Temp
>> pwd

ans=

D:\Temp
```

Operatorii aritmetici de bază în Matlab sunt adunarea (`+`), scăderea (`-`), înmulțirea (`*`), împărțirea (`/`), respectiv ridicarea la putere (`^`). Calculul direct poate fi realizat în linia de comandă:

```
>> 1+2

ans =

     3
```

În general, în cazul în care rezultatul unei operații nu este atribuit niciunei variabile, acesta va fi stocat în variabila specială `ans` (prescurtarea cuvântului *answer*).

În limbajul Matlab, toate variabilele sunt considerate ca fiind de tipul matrice. Cazurile particulare sunt valorile scalare (matrici cu un singur element), respectiv vectorii (matrici unidimensionale), care sunt tratate diferit în anumite situații: de exemplu, o valoare scalară este extinsă în orice dimensiune în cazul operațiilor cu matrici, iar accesul unui element din vector se poate face prin specificarea unui singur indice.

În Matlab, nu este necesară declararea variabilelor. Implicit, valorile numerice sunt de tip *double*, putându-se realiza conversia ulterioară la valori întregi, booleene, etc. Variabilele sunt păstrate în memorie până la terminarea sesiunii Matlab sau până la de-alocarea lor prin comanda `clear`. Inițializarea unei variabile se face utilizând operatorul `=`:

```
>> x = 5

x =

     5
```

Pentru ca rezultatul operației să nu fie afișat în linia de comandă se introduce caracterul `;` la sfârșitul comenzii:

```
>> y = 18;
```

Mai jos sunt prezentate câteva exemple de folosire a operatorilor aritmetici:

```
>> a = (7 * 5 - 3) / 4

a =

     8

>> b = a ^ 3

b =

   512
```

Pentru a inițializa un vector sau o matrice, valorile sunt introduse între paranteze drepte. Elementele de pe aceeași linie sunt separate prin `,` sau spațiu, iar prin `;` se face trecerea la linia următoare. De exemplu, inițializarea unui vector linie se face astfel:

```
>> v = [1 2 3]

v =

     1     2     3
```

Generarea unui vector coloană se face astfel:

```
>> v2 = [1; 2; 3]

v2 =

     1
     2
     3
```

Numărul de elemente dintr-un vector se poate afla utilizând funcția `length`:

```
>> length(v2)

ans =

     3
```

Se pot genera automat vectori linie cu valori specificate într-un interval, prin formatul `a:b:c`; vectorul generat va avea valori între `a` și `c`, cu un increment `b`:

```
>> i = 1:2:9
```

```
i =

     1     3     5     7     9
```

În cazul în care nu este specificat incrementul, acesta este considerat automat ca fiind 1:

```
>> i = 1:4

i =

     1     2     3     4
```

În continuare este prezentat un exemplu de generare a unei matrici de dimensiuni 2×2:

```
>> m = [1 5; 12 9]

m =

     1     5
    12     9
```

Pentru a inițializa o matrice având toate valorile egale cu 0 se poate utiliza funcția Matlab `zeros(m,n)`, unde `m` reprezintă numărul de linii, iar `n` numărul de coloane al matricii:

```
>> z = zeros(2, 3)

z =

     0     0     0
     0     0     0
```

În mod similar, pentru inițializarea unei matrici având toate elementele egale cu 1 se poate utiliza funcția `ones(m,n)`, iar pentru generarea unei matrici cu valori aleatoare distribuite uniform în intervalul $[0, 1]$ se poate utiliza funcția `rand(m,n)`.

Pentru a accesa elemente din vectori și matrici se utilizează paranteze rotunde, ținând cont că în Matlab indexarea elementelor începe cu valoarea 1. Astfel, citirea elementului de pe prima linie și coloana a doua a matricii `m` se face astfel:

```
>> m(1,2)

ans =

     5
```

În cazul vectorilor se poate utiliza un singur indice, fie că este vorba despre un vector line sau unul coloană:

```
>> v2(2)

ans =

     2
```

Există posibilitatea de a accesa mai multe elemente dintr-o matrice utilizând o singură comandă:

```
>> m(2, 1:2)

ans =

    12     9
```

De asemenea, utilizând operatorul `()` se pot introduce elemente în vectori/matrici pe pozițiile specificate:

```
>> m(1, 2) = -6

m =

     1    -6
    12     9
```

Transpusa unei matrici poate fi calculată utilizând operatorul `'`:

```
>> m'

ans =

     1    12
    -6     9
```

Inversa, respectiv determinantul unei matrici se pot calcula utilizând funcțiile Matlab `inv` și `det`:

```
>> inv(m)

ans =

    0.1111    0.0741
   -0.1481    0.0123
```

```
>> det(m)

ans =

    81
```

Operatorii aritmetici Matlab sunt considerați implicit operatori matriciali. Astfel, operatorul `*` reprezintă *produsul matricial*. Pentru realizarea înmulțirii element cu element, se utilizează operatorul `.*` . Care sunt constrângerile impuse operanzilor în cele două cazuri?

```
>> a = [3 5; 8 10]; b = [-5 9; 4 1];
>> a * b

ans =

     5    32
     0    82

>> a .* b

ans =

   -15    45
    32    10
```

În cazul operațiilor cu scalari, aceștia sunt extinși la dimensiunea operanzilor matriciali:

```
>> m = [3 2; 8 10];
>> m + 2

ans =

    5     4
   10    12

>> m * 3

ans =

    9     6
   24    30
```

1.2 Fișiere script și funcții

Comenzile Matlab pot fi introduse atât în linia de comandă, cât și în fișiere script (având extensia .m) ce pot fi apoi rulate. Un fișier script poate conține:

- Fie o listă de comenzi, ce sunt executate secvențial prin introducerea numelui scriptului (fără extensie) în linia de comandă, valorile variabilelor fiind păstrate și după execuția scriptului
- Fie o funcție Matlab, ce poate fi apelată cu parametri și poate întoarce rezultate, variabilele interne fiind dealocate la terminarea execuției funcției

În cazul scripturilor ce conțin o listă de comenzi, se recomandă introducerea la începutul acestora a următoarelor comenzi pentru închiderea tuturor ferestrelor și dealocarea variabilelor, astfel încât rulările anterioare ale scriptului să nu afecteze rezultatele curente:

```
close all;
clear all;
```

Fișierele de tip funcție trebuie să conțină pe prima linie un antet de forma:

```
function [r1, r2, ..., rn] = nume_functie(i1, i2, ..., im)
```

unde `r1, r2, ..., rn` reprezintă variabilele returnate de funcție, argumentele de intrare fiind `i1, i2, ..., im`. Ca excepție, înainte de antet pot fi introduse comentarii, marcate cu simbolul `%`. Aceste comentarii vor apărea ca rezultat al apelării comenzii `help nume_functie`. Uzual, ultima linie de cod a funcției conține cuvântul cheie `return`. Numele fișierului în care este salvată funcția trebuie să fie același cu numele funcției.

Exemplul de mai jos prezintă o funcție Matlab simplă:

```
function [r] = putere(a,b)
  r = a ^ b;
return
```

Apelul funcției din linia de comandă va produce următorul rezultat:

```
>> putere(2,3)

ans =

  8
```

1.3 Structuri de control

Limbajul Matlab pune la dispoziție structurile de control `for`, `while` și `if`, având următoarea sintaxă:

```
for contor = start:increment:stop
     ...
end

while conditie
     ...
end


if conditie1
     ...
elseif conditie2
     ...
else
     ...
end
```

1.4 Grafice

Comanda pentru afișarea graficelor bidimensionale în Matlab este `plot`, aceasta permițând reprezentarea a doi vectori (având același număr de elemente), unul în funcție de celălalt. Un exemplu de afișare a unui grafic este următorul:

```
x = [2 5 9];
y = [6 -1 8];
figure;
plot(x,y);
xlabel('x');
ylabel('y');
title('Exemplu de grafic');
```

Rularea exemplului va genera un grafic similar celui din Figura 1.1. Comanda `figure` deschide o fereastră nouă. În cazul în care această comandă nu este utilizată, comanda `plot` va afișa graficul în fereastra curentă, suprascriind un eventual grafic existent. Axele graficului sunt etichetate utilizând comenzile `xlabel` și `ylabel`, iar titlul poate fi afișat cu comanda `title`. Comanda `plot` dispune de o serie de opțiuni pentru modificarea aspectului graficului, mai multe detalii putând fi obținute prin rularea comenzii `help plot`.

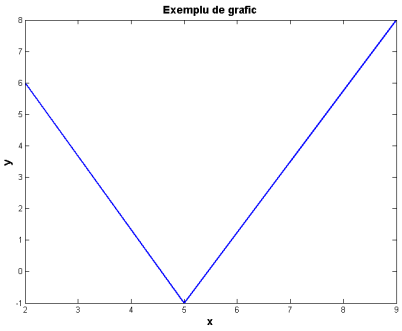


Figura 1.1: Exemplu de grafic.

Pentru a suprapune mai multe grafice se folosește comanda `hold on`, o singură dată după afișarea primului grafic și înaintea afișării următoarelor.

Pentru afișarea mai multor grafice într-o singură fereastră se folosește comanda `subplot`, cu următoarea sintaxă:

```
subplot(m,n,p), plot(x,y);
```

unde `m` și `n` reprezintă numărul de linii, respectiv coloane pe care sunt dispuse graficele, iar `p` reprezintă numărul graficului curent (numerotarea facându-se de la stânga la dreapta și de sus în jos, începând cu 1).

1.5 Generarea semnalelor periodice

Un semnal $x(t)$ se numeşte periodic, de perioadă T, dacă:

$$x(t+T) = x(t), \quad \forall t \in \mathbb{R} \tag{1.1}$$

Dacă $T > 0$ este o perioadă a semnalului $x(t)$, atunci $kT, k \in \mathbb{N}^*$ este de asemenea o perioadă a lui $x(t)$. T se numeşte *perioadă principală* a semnalului $x(t)$ dacă este cea mai mică perioadă.

Un exemplu de semnal periodic este semnalul sinusoidal, descris de formula:

$$x(t) = A\sin(\omega t + \varphi) \tag{1.2}$$

unde A reprezintă amplitudinea semnalului, ω reprezintă frecvența[1] acestuia, iar φ faza semnalului. În domeniul digital, semnalele sunt reprezentate ca șiruri de valori (vectori). Secvența de cod pentru generarea și afișarea formei de undă a unui semnal sinusoidal cu $A = 3$, $\omega = 0.01$ și $\varphi = 10$ este următoarea:

[1] Printr-un abuz de limbaj, vom folosi cuvântul "frecvenţă" pentru a vorbi de ω (pulsaţie sau frecvenţă unghiulară), ştiind că diferenţa faţă de frecvenţa reală este de scalare cu o constantă în radiani : $\omega = 2\pi f$.

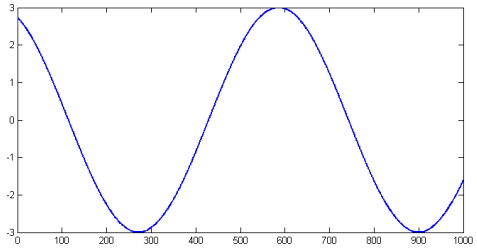


Figura 1.2: Semnal sinusoidal.

```
A = 3;
omega = 0.01;
phi = 10;
t = 1:1000;
x = A*sin(omega*t+phi);
figure;
plot(t,x);
```

Graficul rezultat va fi de tipul celui din Figura 1.2. Pentru generarea semnalului, este necesară crearea prealabilă a unui vector de timp care va conține momentele de timp pentru care sunt generate valorile semnalului. În cazul de față, vectorul de timp `t` conține 1000 de valori, ceea ce înseamnă ca se vor genera 1000 de eșantioane ale semnalului sinusoidal, stocate în vectorul `x`.

Modificați amplitudinea, frecvența, faza, respectiv numărul de valori generate ale semnalului. Ce observați?

Alte semnale periodice, cum ar fi cele de tip dinte de fierăstrău și dreptunghiular pot fi generate în mod similar exemplului precedent, înlocuind funcția `sin` cu `sawtooth`, respectiv `square`. Pentru generarea acestor semnale utilizând mediul Octave, este necesară încărcarea prealabilă a pachetului *signal*, aceasta facându-se prin introducerea următoarei comenzi:

```
>> pkg load signal
```

1.6 Generarea unor semnale aleatoare

Semnalele aleatoare sunt generate ca vectori cu valori aleatoare, utilizând funcțiile Matlab `rand` (pentru semnale cu distribuție uniformă) sau `randn` (distribuție normală sau Gaussiană). De exemplu, secvența de cod următoare generează 1000 de eșantioane ale unui semnal aleator cu distribuție uniformă (vezi Figura 1.3):

```
n = 1000;
x = rand(1,n);
```

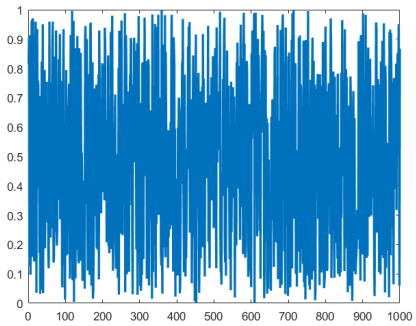


Figura 1.3: Semnal aleator cu distribuție uniformă.

Problema generării semnalelor aleatoare va fi abordată pe larg în cadrul Lucrării 7.

1.7 Exerciții

1. Să se implementeze o funcție `fibo` care calculează recursiv elementul de pe poziția n din şirul lui Fibonacci, definit cu formula:

$$fibo[n] = \begin{cases} 1 & \text{dacă } n <= 2, \\ fibo[n-1] + fibo[n-2] & \text{altfel.} \end{cases}$$

   Să se implementeze o funcţie care generează în mod iterativ un vector conținând primele n elemente din şirul lui Fibonacci.

2. Să se genereze 600 de eșantioane ale unui semnal sinusoidal $x(t)$, de amplitudine $A = 2$, frecvență $\omega = 0.02$ și fază $\varphi = 0$. Să se genereze același număr de eșantioane ale unui semnal sinusoidal $y(t)$, cu frecvență dublă față de cea a semnalului $x(t)$. Să se genereze semnalul $z(t)$ ce reprezintă suma celor două semnale.

   Să se afișeze formele de undă ale celor trei semnale în aceeași fereastră, una sub alta. Modificați semnalul $y(t)$ astfel încât să aibă amplitudine dublă față de $x(t)$. Ce observați?

3. Să se genereze 2000 de eșantioane ale unui semnal sinusoidal $x(t)$, de amplitudine $A = 1$, frecvență $\omega = 0.01$ și fază $\varphi = 0$. Să se genereze același număr de eșantioane ale unui semnal *cosinusoidal* $y(t)$, cu o frecvență de 10 ori mai mare față de cea a semnalului $x(t)$. Să se genereze semnalul $z(t)$ rezultat din modularea în amplitudine a semnalului $x(t)$ cu purtătoarea $y(t)$, utilizând formula:

$$z(t) = (1 + k \cdot x(t)) \cdot y(t)$$

unde gradul de modulație $k = 0.5$.

Să se afișeze formele de undă ale celor trei semnale în aceeași fereastră, una sub alta. Modificați valoarea gradului de modulație. Ce observați?

4. Să se genereze $N = 1000$ de eșantioane ale unui semnal sinusoidal $x(t)$, de amplitudine $A = 3$, frecvență $\omega = 0.01$ și fază $\varphi = 0$. Să se calculeze energia acestui semnal, pe baza formulei:

$$E = \sum_{i=1}^{N} |x[i]|^2 \tag{1.3}$$

Pentru calculul modulului utilizați funcția Matlab `abs`. Calculați energia unui semnal de tip dinți de fierăstrău, respectiv a unui semnal dreptunghiular.

5. Să se genereze 300 de eșantioane ale unui semnal de tip "chirp", a cărui frecvență crește liniar în timp (vezi Figura 1.4). Pentru generarea semnalului, utilizați funcția Matlab următoare:

```
chirp(t, w1, t1, w2);
```

unde `t` reprezintă vectorul cu valorile de timp, `w1` reprezintă frecvența inițială, stabilă până la momentul de timp `t1`, în timp ce `w2` reprezintă frecvența finală.

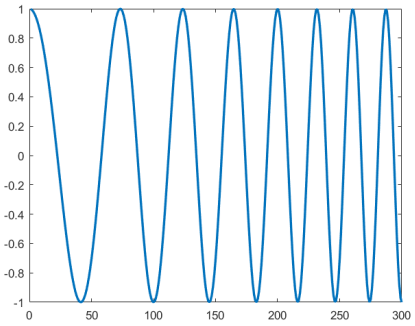


Figura 1.4: Semnal de tip "chirp".

Lucrarea 2

Serii Fourier

Problematica studiată în această lucrare constă în generarea unei baze de funcții reale ortogonale alcătuită din funcții de tip sinus și cosinus pentru reprezentarea semnalelor, precum și descompunerea semnalelor în serie Fourier trigonometrică.

2.1 Serii Fourier trigonometrice

Spațiul Hilbert al funcțiilor reale este descris de $(H; <\cdot,\cdot>, ||\cdot||)$, unde H reprezintă mulțimea funcțiilor reale, continue, derivabile, $<\cdot,\cdot>$ reprezintă produsul scalar, iar $||x|| = \sqrt{<x,x>}$ reprezintă norma spațiului. De exemplu, în cazul funcțiilor continue, derivabile o dată, definite pe intervalul $[a,b]$, produsul scalar se poate defini astfel:

$$< f,g >= \int_a^b f(t)\cdot g(t)\mathrm{d}t \tag{2.1}$$

Produsul scalar reprezintă proiecția funcției f pe g, rezultatul operației indicând măsura în care una din funcții o conține pe cealaltă. În cazul discret, pentru vectori de N eșantioane, produsul scalar se poate defini ca:

$$< f,g >= \sum_{i=1}^{N} f(i)\cdot g(i) \tag{2.2}$$

O bază de funcții $B = \{e_i | i \in \mathbb{N}\}, B \subset H$ se numeşte *sistem ortogonal* dacă funcțiile ce o alcătuiesc îndeplinesc următoarea condiție:

$$< e_i, e_j >= \begin{cases} k & ,i = j \\ 0 & ,i \neq j \end{cases} \tag{2.3}$$

În cazul în care $k = 1$, B se numeşte *sistem ortonormal* sau *ortonormat*:

$$< e_i, e_j >= \delta_{ij} = \begin{cases} 1 & , i = j \\ 0 & , i \neq j \end{cases} \tag{2.4}$$

Orice semnal periodic $s(t)$, de perioadă principală T, poate fi scris sub forma:

$$s(t) = c_0 + \sum_{n \geq 1} c_n \cos(n\omega_0 t) \ + \sum_{n \geq 1} s_n \sin(n\omega_0 t) \tag{2.5}$$

unde $\omega_0 = \frac{2\pi}{T}$ se numește frecvență fundamentală.

Relația (2.5) reprezintă dezvoltarea semnalului $s(t)$ în *serie Fourier trigonometrică*, valoarea c_0 reprezentând componenta continuă a semnalului, în timp ce restul reprezintă componentele armonice. Conform relației, orice semnal periodic poate fi descompus într-o sumă de sinusoide și cosinusoide având frecvențe multipli întregi ai frecvenței fundamentale a semnalului.

Funcțiile $\cos(n\omega_0 t)$, $n \in \mathbb{N}$ și $\sin(n\omega_0 t)$, $n \in \mathbb{N}^*$, formează un sistem ortogonal de funcții. Considerând produsul scalar a două funcții f și g periodice, de perioadă T, ca fiind:

$$< f, g >= \int_T f(t) \cdot g(t) dt \tag{2.6}$$

se poate arăta că:

$$< \cos(m\omega_0 t), \cos(n\omega_0 t) >= \begin{cases} T, m = n = 0 \\ \frac{T}{2}, m = n \neq 0 \\ 0, m \neq n \end{cases} \tag{2.7}$$

$$< \sin(m\omega_0 t), \sin(n\omega_0 t) >= \begin{cases} \frac{T}{2}, m = n \neq 0 \\ 0, m \neq n \end{cases} \tag{2.8}$$

$$< \sin(m\omega_0 t), \cos(n\omega_0 t) >= 0, \forall m, n \tag{2.9}$$

Relațiile de mai sus permit calculul coeficienților c_n și s_n ai dezvoltării semnalului $s(t)$ în serie Fourier trigonometrică, rezultând:

$$\begin{cases} c_0 = \frac{1}{T} < s(t), 1 >= \frac{1}{T} \int\limits_T s(t)\ \mathrm{d}t \\ \\ c_n = \frac{2}{T} < s(t), \cos(n\omega_0 t) >= \frac{2}{T} \int\limits_T s(t) \cdot \cos(n\omega_0 t)\ \mathrm{d}t \\ \\ s_n = \frac{2}{T} < s(t), \sin(n\omega_0 t) >= \frac{2}{T} \int\limits_T s(t) \cdot \sin(n\omega_0 t)\ \mathrm{d}t \end{cases} \tag{2.10}$$

2.2 Desfășurarea lucrării

În cadrul acestei lucrări se va genera un set de semnale sinusoidale, având frecvențe multipli ai unei frecvențe de bază, se va calcula produsul scalar al acestora și se vor analiza rezultatele obținute.

Să se genereze $N = 80$ eșantioane ale semnalelor următoare (Figura 2.1):

- $s_1(t) = \sin(\omega t)$
- $s_2(t) = \sin(2\omega t)$
- $s_3(t) = \sin(3\omega t)$

cu frecvența de bază $\omega = \frac{\pi}{40}$ (perioada va fi $T = \frac{2\pi}{\omega} = 80$).

Secvența Matlab pentru generarea și afișarea celor trei semnale este următoarea:

```
omega = pi/40;
N = 80;
t = 0:N-1;

s1 = sin(omega * t);
subplot(3, 1, 1), plot(t, s1);

s2 = sin(2 * omega * t);
subplot(3, 1, 2), plot(t, s2);

s3 = sin(3 * omega * t);
subplot(3, 1, 3), plot(t, s3);
```

Să se calculeze produsul scalar $< s_1(t), s_1(t) >$, conform formulei (2.2). Secvența pentru realizarea calculului este prezentată mai jos, rezultatul fiind afișat în linia de comandă:

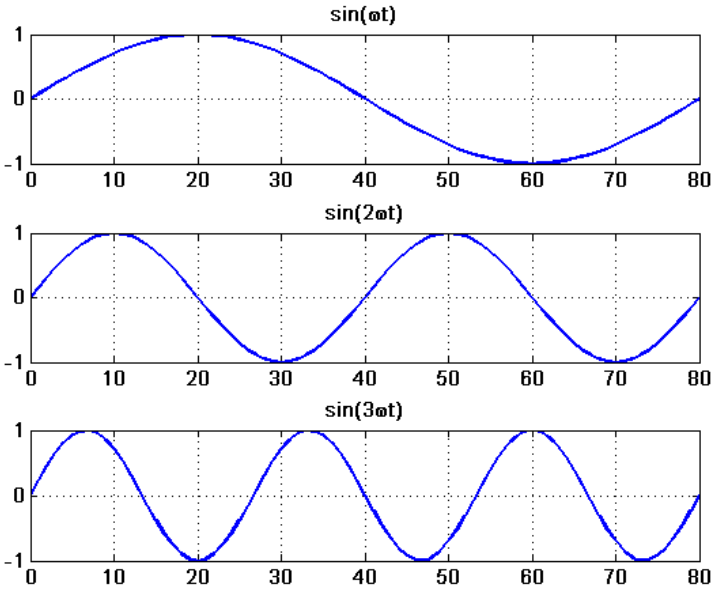


Figura 2.1: Semnalele $\sin(\omega t)$, $\sin(2\omega t)$, $\sin(3\omega t)$.

```
s1s1 = 0;
for i = 1 : N
    s1s1 = s1s1 + s1(i) * s1(i);
end

fprintf('<s1, s1> = %f\n', s1s1);
```

Să se calculeze produsul scalar $< s_1(t), s_2(t) >$, folosind următoarea secvență:

```
s1s2 = 0;
for i = 1 : N
    s1s2 = s1s2 + s1(i) * s2(i);
end

fprintf('<s1, s2> = %f\n', s1s2);
```

Se poate observa că rezultatele obținute confirmă proprietățile de ortogonalitate enuntațe în (2.8). În mod similar exemplelor de mai sus, calculați produsele scalare $< s_1(t), s_3(t) >$, respectiv $< s_2(t), s_3(t) >$. Cum se poate realiza calculul fără utilizarea unei structuri `for`?

2.3 Exercițiu

Să se genereze și afișeze $N = 80$ eșantioane ale semnalului reprezentat în Figura 2.2:

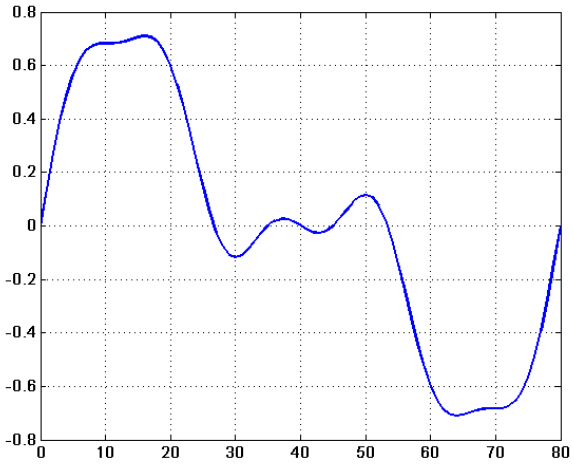


Figura 2.2: Semnal compus din trei sinusoide.

$$s(t) = 0.5\sin(\omega t) + 0.4\sin(2\omega t) + 0.1\sin(5\omega t), \quad \omega = \frac{\pi}{40} \tag{2.11}$$

Să se genereze câte N eșantioane pentru 5 semnale de forma :

$$s_n(t) = sin(n\omega t), \quad n = 1, 2, ..., 5 \tag{2.12}$$

1. Calculați produsele scalare $< s(t), s_1(t) >$, $< s(t), s_2(t) >$, ..., $< s(t), s_5(t) >$.
2. Calculați coeficienții s_1, s_2, ..., s_5 ai dezvoltării în serie Fourier trigonometrică a semnalului $s(t)$.

Lucrarea 3

Transformata Fourier

În cadrul acestei lucrări vor fi studiate spectrele Fourier de amplitudine ale câtorva semnale de interes, calculate cu ajutorul transformatei Fourier rapide, disponibilă ca funcție Matlab.

3.1 Relațiile de transformare

Spre deosebire de seriile Fourier, transformata Fourier poate fi utilizată pentru analiza semnalelor *neperiodice*. Fie un semnal $x(t)$ de modul integrabil:

$$\int_{-\infty}^{+\infty} x(t)dt = M < \infty \tag{3.1}$$

Transformata Fourier a semnalului $x(t)$ este definită astfel:

$$X(\omega) = \int_{-\infty}^{+\infty} x(t)e^{-j\omega t}dt \tag{3.2}$$

Transformata Fourier $X(\omega)$ reprezintă o funcție complexă, ce poate fi scrisă în funcție de modul și fază în felul următor:

$$X(\omega) = |X(\omega)|e^{+j\varphi\omega} \tag{3.3}$$

Astfel, modulul $|X(\omega)|$ reprezintă amplitudinea, iar $\varphi(\omega)$ faza cosinusoidei complexe de frecvență ω ce face parte din descompunerea spectrală a semnalului $x(t)$. Toate valorile $|X(\omega)|$ alcătuiesc *spectrul de amplitudini* al semnalului $x(t)$, iar fazele alcătuiesc *spectrul de faze*. Dacă $x(t)$ este un semnal cu valori reale, atunci transformata Fourier va fi simetrică față de axa verticală, valorile transformatei Fourier fiind situate simetric față de zero și

complex conjugate (astfel apar *frecvențele imagine*, cu semn negativ, care nu există în realitate în spectrul semnalului):

$$X(-\omega) = X^*(\omega) \Leftrightarrow \begin{cases} |X(-\omega)| = |X(\omega)| \\ \varphi(-\omega) = -\varphi(\omega) \end{cases} \tag{3.4}$$

Relația inversă de transformare, care permite recuperarea semnalului $x(t)$ din spectrul $X(\omega)$ este:

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} X(\omega) e^{+j\omega t} d\omega \tag{3.5}$$

În Figura 3.1 este prezentat un semnal cosinusoidal de frecvență ω_0 și spectrul său de amplitudini teoretic. Pe lângă componenta de frecvență ω_0, se poate constata prezența în spectru a componentei corespunzătoare frecvenței imagine $-\omega_0$.

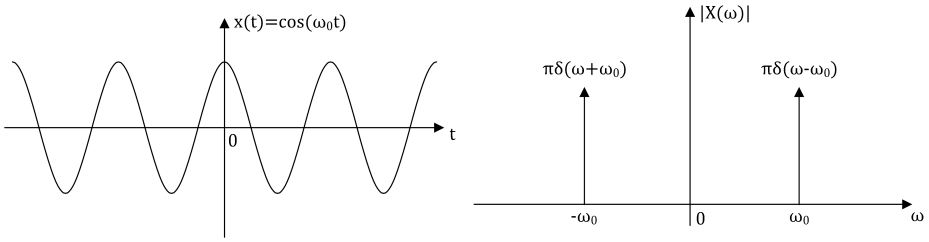


Figura 3.1: Semnal cosinusoidal (a) și spectrul său de amplitudini (b).

Varianta utilizată în analiza semnalelor discretizate în timp este Transformata Fourier Discretă (Discrete Fourier Transform - DFT). În general, în implementările digitale se utilizează o variantă eficientă de calcul a acesteia, denumită Transformata Fourier Rapidă (Fast Fourier Transform - FFT).

3.2 Desfășurarea lucrării

Să se genereze un semnal cosinusoidal. Să se calculeze și afișeze spectrul Fourier de amplitudini al acestui semnal. Generarea și afișarea a $N = 300$ eșantioane ale semnalului cosinusoidal `x`, de amplitudine $A = 1$ și frecvență $\omega = 0.2$ se face utilizând secvența de mai jos:

```
N = 300;
t = 1:N;
omega = 0.2;
x = cos(omega*t);
```

```
figure;
plot(t, x);
```

Calculul și afișarea spectrului de amplitudini se face astfel:

```
X = fft(x);
w = -pi : (2*pi)/N : pi - (2*pi)/N;
figure;
plot(w, abs(fftshift(X)));
```

Funcția `fft` realizează calculul transformatei Fourier rapide, în timp ce funcția `fftshift` se utilizează pentru a avea componenta continuă ($\omega = 0$) în centrul graficului spectrului, în timp ce `abs` calculează valoarea absolută a argumentului. De remarcat este faptul că componentele reprezentate utilizând transformata Fourier rapidă sunt în număr de N (egal cu numărul de eșantioane ale semnalului analizat), distanțate egal în intervalul $[-\pi, \pi)$.

Repetați pașii de generare a semnalului și calcul al transformatei Fourier pentru diverse frecvențe ale semnalului `x`.

3.3 Exerciții

1. Să se calculeze și afișeze spectrul Fourier de amplitudini pentru următoarele semnale:
   - Impuls unitate
   - Semnal constant
   - Impuls dreptunghiular
2. Să se verifice proprietatea de liniaritate a transformatei Fourier:

$$\mathcal{F}\{ax(t) + by(t)\}(\omega) = a\mathcal{F}\{x(t)\}(\omega) + b\mathcal{F}\{y(t)\}(\omega) \tag{3.6}$$

   - Să se genereze două cosinusoide, $x(t)$ de frecvență $\omega_x = 0.2$ și $y(t)$ de frecvență $\omega_y = 0.3$
   - Să se calculeze și afișeze suma spectrelor de amplitudine ale celor două cosinusoide
   - Să se calculeze și afișeze spectrul semnalului rezultat prin însumarea celor două cosinusoide

Lucrarea 4

Eșantionarea semnalelor

În cadrul acestei lucrări ne vom concentra atenţia asupra câtorva aspecte, atât teoretice cât şi practice şi de interpretare, ale eşantionării, aceasta fiind o primă etapă în conversia semnalelor din domeniul analogic în cel digital.

4.1 Teorema eșantionării

Teorema eşantionării, cunoscută și ca teorema Nyquist-Shannon, este de o deosebită importanţă pentru trecerea de la domeniul funcţiilor sau al semnalelor continue la domeniul funcţiilor şi al semnalelor discrete, practic pentru conversia semnalelor analogice în semnale digitale. Discretizarea semnalelor se face *în timp* (proces ce poartă numele de *eşantionare*) şi *în valoare* (proces ce poară numele de *cuantizare*). Din punct de vedere ingineresc, eşantionarea reprezintă procesul convertirii unui semnal, a unei funcţii *continue* de timp, într-o secvenţă sau şir de valori, adică într-o funcţie *discretă* de timp.

Fie $x(t)$ un semnal de modul integrabil, al cărui spectru $X(\omega)$ este dat de transformata Fourier:

$$X(\omega) = \mathcal{F}\{x(t)\}(\omega) = \int_{-\infty}^{+\infty} x(t)e^{-j\omega t}dt \tag{4.1}$$

Semnalul $x(t)$ poate fi recuperat din spectrul $X(\omega)$ prin transformată Fourier inversă:

$$x(t) = \mathcal{F}^{-1}\{X(\omega)\}(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} X(\omega)e^{+j\omega t}d\omega \tag{4.2}$$

Vom presupune în continuare că spectrul semnalului este mărginit, de forma celui din Figura 4.1. Mărginirea spectrului se exprimă matematic ca:

$$|X(\omega)| = 0 \quad \forall \omega > \Omega_{max}, \tag{4.3}$$

unde Ω_{max} este frecvenţa maximă din spectrul semnalului.

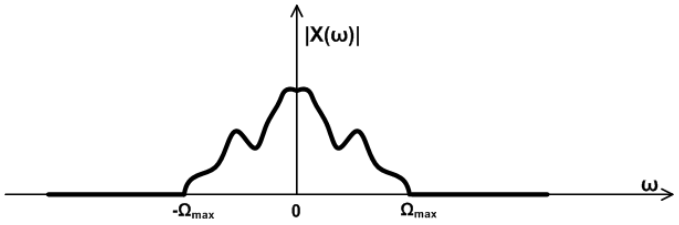


Figura 4.1: Exemplu de spectru mărginit.

Având în vedere condiţiile descrise anterior, enunţul teoremei eşantionării este următorul: *Un semnal $x(t)$ de spectru mărginit poate fi complet reconstruit din eşantioanele sale, cu condiţia ca frecvenţa de eşantionare Ω_e să fie cel puţin dublul frecvenţei maxime Ω_{max} din spectrul semnalului.*

$$x(t) = \sum_{k=-\infty}^{+\infty} x(kT_e) sinc(\Omega_{max}(t - kT_e)), \quad \forall t \in \mathbb{R} \tag{4.4}$$

$x(kT_e)$ reprezintă eşantioane prelevate din semnalul original, la momente de timp kT_e (multipli întregi de perioada de eşantionare), egal distanţate. Acesta reprezintă cazul eşantionării *uniforme*.

Versiunea originală a teoremei, care aparţine lui Shannon, este următoarea: *Dacă o funcţie $x(t)$ nu conţine frecvenţe mai mari decât B Hz, atunci ea este complet determinată de valorile funcţiei într-o serie de puncte distanţate la $\frac{1}{2B}$ secunde.*

Condiţia ca frecvenţa de eşantionare să fie măcar dublul frecvenţei maxime din spectrul semnalului poartă numele de *condiţie Nyquist*.

$$\Omega_e = \frac{2\pi}{T_e} = \frac{2\pi}{\frac{\pi}{\Omega_{max}}} = 2\Omega_{max} \tag{4.5}$$

$$T_e = \frac{2\pi}{\Omega_e} = \frac{\pi}{\Omega_{max}} \tag{4.6}$$

Vom reproduce aici demostraţia originală a lui Shannon, extrem de elegantă, dar mai puţin intuitivă pentru înţelegerea refacerii semnalului prin interpolarea utilizând funcţii sinus cardinal, mai concret o filtrare trece-jos (vezi Lucrarea 6).

Demonstraţie: Fie $X(\omega)$ spectrul Fourier al semnalului $x(t)$. Atunci:

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} X(\omega) e^{+j\omega t} d\omega = \frac{1}{2\pi} \int_{-2\pi W}^{+2\pi W} X(\omega) e^{+j\omega t} d\omega$$

deoarece $X(\omega)$ este zero în afara intervalului (benzii) de frecvenţe $[-2\pi W, 2\pi W]$, unde $\Omega_{max} = 2\pi W$.

Dacă $t = \frac{n}{2W}$, unde n este un întreg pozitiv (sau negativ), atunci obţinem:

$$x\left(\frac{n}{2W}\right) = \frac{1}{2\pi} \int_{-2\pi W}^{+2\pi W} X(\omega) e^{+j\omega \frac{n}{2W}} d\omega$$

În partea stângă nu avem altceva decât valorile eşantioanelor prelevate din $x(t)$. Integrala din partea dreaptă nu este altceva decât coeficientul n al dezvoltării în serie Fourier a funcţiei $X(\omega)$, considerând intervalul $[-2\pi W, 2\pi W]$ ca perioadă fundamentală. Dar funcţia $x(t)$ poate fi refăcută oricând din spectrul $X(\omega)$, prin urmare ea poate fi refăcută din eşantioanele sale.

Eşantioanele semnalului se obţin prin înmulţirea semnalului cu funcţia de eşantionare de tip "pieptene" de eşantionare (Figura 4.2), formată dintr-o sumă sau succesiune de impulsuri Dirac, egal distanţate în timp cu perioada de eşantionare.

$$\delta_{T_e}(t) = \sum_{n \in \mathbb{Z}} \delta(t - nT_e) \tag{4.7}$$

Figura 4.2: Funcţia "pieptene" de eşantionare.

Consecinţa eşantionării o reprezintă periodizarea spectrului semnalului original cu Ω_e (vezi Figura 4.3), datorită descompunerii în serie Fourier a spectrului semnalului original folosită în demostraţia teoremei.

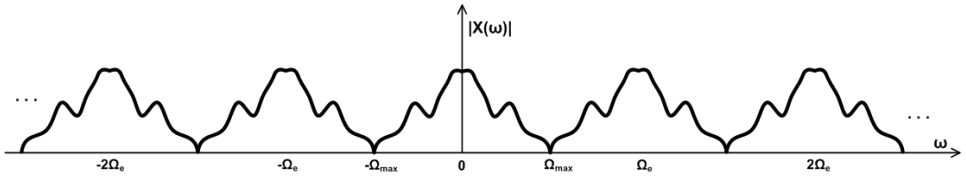


Figura 4.3: Periodizarea spectrului.

Conform teoremei, reconstrucţia semnalului este realizată ca o interpolare, ca o sumă de funcţii sinus cardinal ponderate cu valorile eşantioanelor $x(kT_e)$ ale semnalului. Relaţia (4.4) poate fi privită şi ca o descompunere a semnalului original $x(t)$ într-o bază ortonormată de funcţii $\{f_k(t)\}_{k \in \mathbb{Z}}$ (vezi Figura 4.4), unde:

$$f_k(t) = sinc(\Omega_{max}(t - kT_e)) \tag{4.8}$$

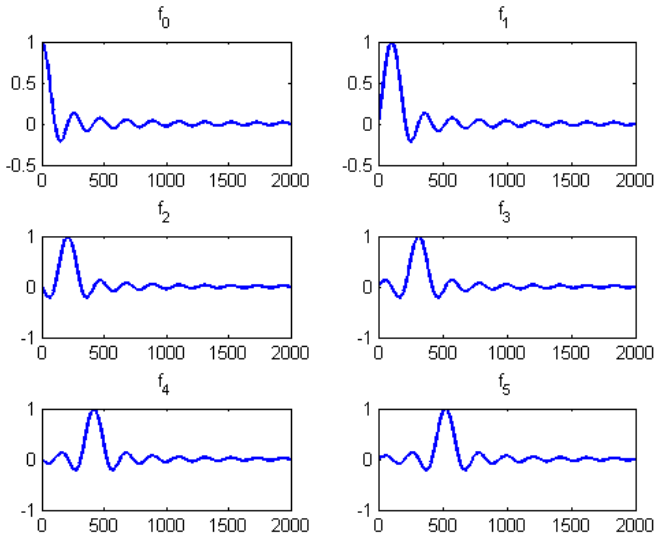


Figura 4.4: Baza de funcţii sinus cardinal pentru $k = \overline{0..5}$.

În funcţie de frecvenţa Ω_e cu care se realizează eşantionarea, distingem trei cazuri:

- eşantionare *corectă*, atunci când $\Omega_e = 2\Omega_{max}$,
- *supraeşantionare*, pentru $\Omega_e > 2\Omega_{max}$ şi
- *subeşantionare*, când nu este îndeplinită conditţa Nyquist, $\Omega_e < 2\Omega_{max}$.

În cazul supraeşantionării se realizează o separare mai bună între replicile spectrale, iar semnalul poate fi refăcut corect, la fel ca în cazul unei eşantionări corecte. În cazul subeşantionării, însă, apare fenomenul nedorit de "aliere" (*eng. aliasing*) adică de suprapunere a replicilor spectrale. Aceast fenomen are drept urmare imposibilitatea refacerii corecte a semnalului original din eşantioanele prelevate cu $\Omega_e < 2\Omega_{max}$. Spectrul semnalului refăcut va conţine frecvenţe aparent înalte dar care nu sunt altceva decât frecvenţe joase din prima replică spectrală, amestecate (aliate) cu cele din spectrul de bază al semnalului (vezi Figura 4.5).

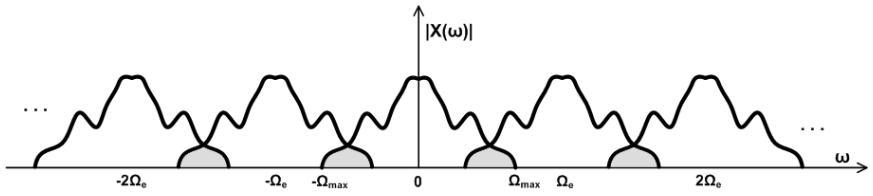


Figura 4.5: Fenomenul de aliere a replicilor spectrale.

4.2 Desfăşurarea lucrării

În cadrul acestei lucrări se va genera un semnal format dintr-o sumă de sinusoide, se va eşantiona cu diverse frecvenţe de eşantionare, se va reface semnalul folosind formula dată de teorema eşantionării şi se vor analiza rezultatele obţinute.

Să se genereze un semnal de forma:

$$x(t) = A_1 \cdot sin(\omega_1 t + \varphi_1) + A_2 \cdot sin(\omega_2 t + \varphi_2) + A_3 \cdot sin(\omega_3 t + \varphi_3) \quad (4.9)$$

De exemplu, dacă $A_1 = 1$, $\omega_1 = \omega = 0.01$, $\varphi_1 = 0$, $A_2 = 0.5$, $\omega_2 = 2\omega$, $\varphi_2 = 10$ şi $A_3 = 0.25$, $\omega_3 = 3\omega$, $\varphi_3 = 30$ se obţine semnalul din Figura 4.6.

Secvenţa Matlab pentru generarea acestui semnal este următoarea:

```
N = 2000;
omega = 0.01;
t = 1:N;
x = sin(omega*t) + 0.5*sin(2*omega*t+10) + 0.25*sin(3*omega*t+30);
figure;
plot(t,x);
```

Să se eşantioneze acest semnal cu $\Omega_e = 2\Omega_{max} = 6\omega$, şi să se reprezinte grafic eşantioanele folosind funcţia Matlab `stem`.

```
Te = floor( 2*pi / (6*omega) );
t_es = 1:Te:N;
x_es = x( t_es );
hold on;
stem( t_es, x_es );
```

Să se reconstituie semnalul original, din eşantioanele prelevate, folosind interpolarea cu funcţii sinus cardinal ($sinc(x) = \frac{sin(x)}{x}$):

```
for i = 1:N
   s = 0;
   for k = 1:length( t_es )
```

```
      s = s + x_es(k) * sinc((3*omega*(i-t_es(k)))/pi);
   end;
   x_rec(i) = s;
end;

plot( t, x_rec, 'r-' );
```

Rezultatul ar trebui să semene cu cel din Figura 4.6. Puteţi observa cum în cazul eşantionării corecte cu $\Omega_e = \Omega_{max}$ semnalul reconstruit (roşu) din eşantioanele prelevate (negru) este aproape identic cu semnalul original (albastru).

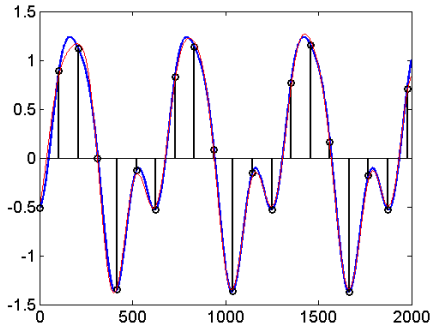
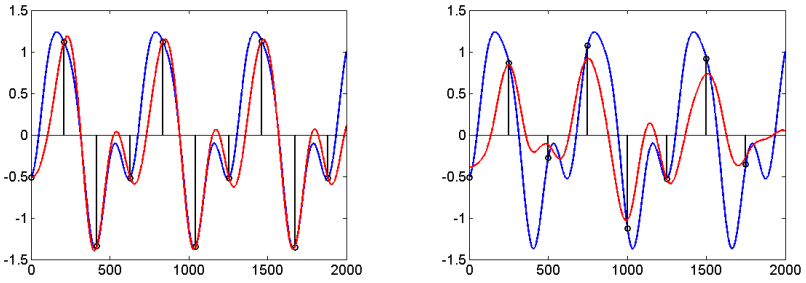


Figura 4.6: Semnalul $x(t)$ original, eşantionat şi reconstruit, în cazul eşantionării corecte cu $\Omega_e = 2\Omega_{max} = 6\omega$.

Observaţie: Dacă semnalul original era reprezentat în 2000 de puncte, acesta poate fi reconstruit din doar 20 de eşantioane.

Repetaţi procedeul (eşantionare şi reconstrucţie) pentru $\Omega_e = \Omega_{max} = 3\omega$ şi $\Omega_e = \frac{\Omega_{max}}{2} = 1.5\omega$. Observaţi efectele subeşantionării. Semnalul original nu mai poate fi refăcut din eşantioanele prelevate, dacă nu a fost respectată condiţia Nyquist (vezi Figura 4.7).



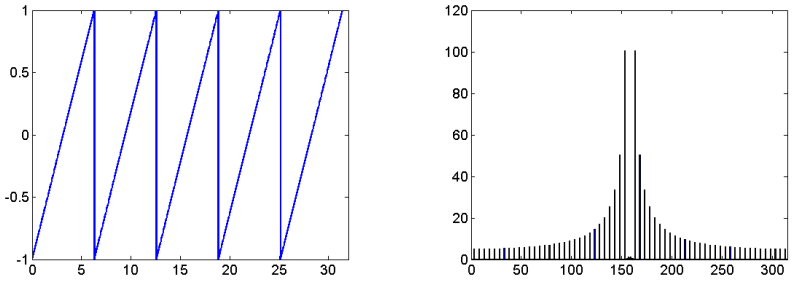
(a) (b)

Figura 4.7: Semnalul $x(t)$ original, eşantionat şi reconstruit în cazul unei subeşantionări cu (a) $\Omega_e = 3\omega$ şi (b) $\Omega_e = 1.5\omega$.

4.3 Exerciții

1. Să se reia exemplul din desfăşurarea lucrării pentru diverse frecvenţe de eşantionare ce nu respectă condiţia Nyquist. Observaţi efectele atât asupra formei semnalului reconstruit, cât şi asupra spectrului Fourier de amplitudine al acestuia, comparativ cu semnalul original.

2. Să se genereze un semnal de tip dinte de fierăstrău, folosind funcţia Matlab `sawtooth`. Să se repete paşii din desfăşurarea lucrării, alegând Ω_{max} frecvenţa din spectrul semnalului pentru care $|X(\omega)|$ reprezintă aproximativ 10% din valoarea maximă a spectrului de amplitudine. Ce observaţi?



(a) (b)

Figura 4.8: Semnalul dinte de fierăstrău şi spectrul său Fourier de amplitudine.

Lucrarea 5

Sisteme de timp discret

În cadrul acestei lucrări vor fi studiate o serie de sisteme dedicate prelucrării secvențelor discrete de date. În urma discretizării în timp a unui semnal continuu $x(t)$, se obține secvența de eșantioane (sau semnalul discret) $x[n], n \in \mathbb{Z}$. Funcția sistemelor de timp discret este de a prelucra o secvență de intrare $x[n]$ pentru a genera o secvență de ieșire $y[n]$. Acestea reprezintă sisteme de tip single-input single-output (SISO) (Figura 5.1). În cele ce urmează vor fi studiate patru sisteme de timp discret: acumulatorul, derivatorul de ordinul unu, filtrul de mediere și interpolatorul liniar.

Figura 5.1: Sistem de timp discret.

5.1 Acumulatorul

Acumulatorul reprezintă un sistem a cărui ieșire la un anumit moment de timp reprezintă suma tuturor eșantioanelor de la intrare de până la momentul respectiv. Sistemul este descris de următoarele relații între intrare și ieșire:

$$y[n] = \sum_{k=1}^{n} x[k], n = 1, ..., N \tag{5.1}$$

$$\begin{aligned} &y[1] = x[1] \\ &y[n] = y[n-1] + x[n], n = 2, ..., N \end{aligned} \tag{5.2}$$

Cele două variante sunt echivalente. În varianta (5.1), ieșirea la momentul n este calculată pe baza tuturor valorilor de intrare de până la momentul n,

în timp ce în varianta (5.2) este una recursivă, ieșirea curentă fiind calculată pe baza intrării curente și a *ieșirii* precedente (implementare cu reacție - *feedback*).

Să se genereze un semnal ce conține un impuls unitate și să se aplice la intrarea unui acumulator implementat conform variantei (5.1) (vezi Figura 5.2 (a) și (b)).

Generarea și afișarea a $N = 100$ eșantioane ale semnalului de intrare se face astfel:

```
N = 100;
x = zeros(1,N);
x(N/2) = 1;

figure;
stem(x);
```

Următoarea secvență realizează implementarea acumulatorului conform formulei (5.1) și afișarea semnalului de ieșire:

```
y = zeros(1,N);
for n = 1:N
    for k = 1:n
        y(n) = y(n) + x(k);
    end
end

figure;
stem(y);
```

Exerciții

1. Înlocuiți semnalul de la intrare cu o secvență aleatoare, generată utilizând funcția Matlab `rand` (vezi Figura 5.2 (c) și (d)).

2. Implementați acumulatorul utilizând relația (5.2).

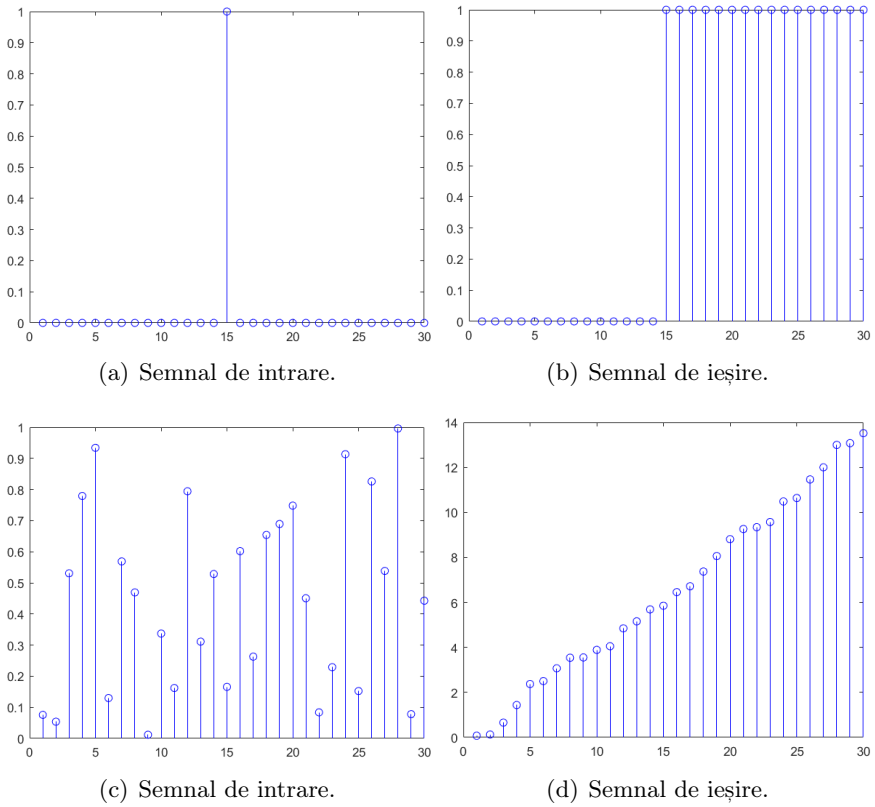
5.2 Derivatorul de ordinul unu

Derivatorul de ordinul unu reprezintă practic un filtru trece-sus, având rolul de a scoate în evidență tranzițiile bruște din vectorul reprezentând semnalul discret de prelucrat. Derivatorul este definit prin următoarele relații între intrare și ieșire:

$$y[n] = x[n] - x[n-1], n = 2, ..., N \tag{5.3}$$

$$y[n] = x[n+1] - x[n], n = 1, ..., N-1 \tag{5.4}$$

Cele două relații sunt echivalente, însă doar varianta (5.3) poate fi utilizată într-o implementare ce poate opera în timp real; în cazul variantei (5.4), pentru calculul valorii de ieșire la momentul n este necesară cunoașterea valorii intrării de la momentul de timp următor, $n+1$, valoare necunoscută în cazul unui sistem de timp real. Acest fapt nu constituie un impediment atunci când întregul semnal de intrare este disponibil pentru procesare.



(a) Semnal de intrare.

(b) Semnal de ieșire.

(c) Semnal de intrare.

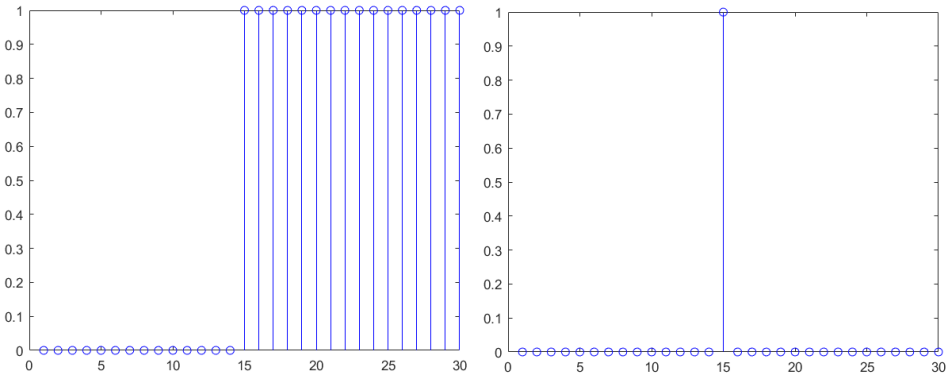
(d) Semnal de ieșire.

Figura 5.2: Semnalele de la intrarea (a), (c) și ieșirea (b), (d) acumulatorului.

Exerciții

1. Să se genereze $N = 100$ eșantioane ale unui semnal discret de tip treaptă unitate și să se aplice la intrarea unui derivator implementat conform uneia din variantele (5.3) sau (5.4) (vezi Figura 5.3).

2. Pornind de la una din ecuațiile (5.3) sau (5.4), să se deducă relația dintre intrare și ieșire pentru un derivator de ordinul doi și să se implementeze acest sistem. Aplicați semnalul de tip treaptă unitate la intrarea derivatorului de ordin doi, precum și la intrarea unui sistem alcătuit din două derivatoare de ordinul unu conectate în cascadă.



(a) Semnal de intrare. (b) Semnal de ieșire.

Figura 5.3: Intrarea (a) și ieșirea (b) unui derivator de ordinul unu.

5.3 Filtrul de mediere

Filtrul de mediere reprezintă un filtru trece-jos, utilizat pentru reducerea (netezirea) variațiilor bruște dintr-un semnal. Valoarea unui eșantion de la ieșire este media a M eșantioane consecutive din semnalul de intrare.

Relația dintre intrare și ieșire pentru un filtru de mediere cu fereastră de M eșantioane este următoarea:

$$y[n] = \frac{1}{M} \sum_{k=0}^{M-1} x[n-k], n = M, ..., N \tag{5.5}$$

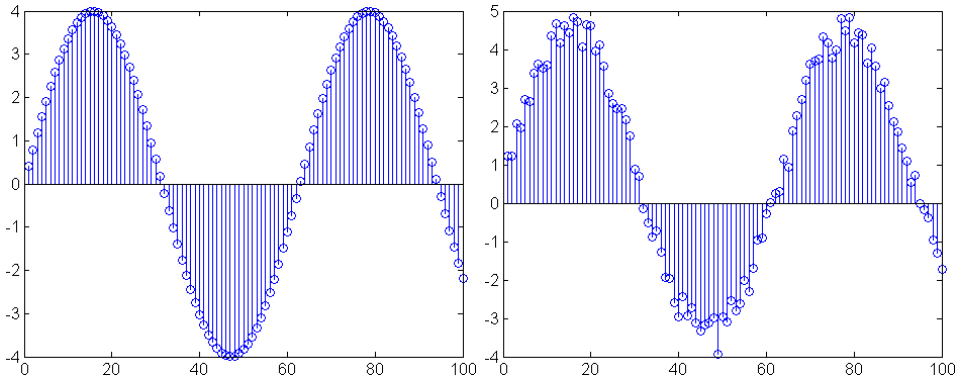
Exercițiu

Să se genereze $N = 100$ eșantioane ale unui semnal sinusoidal de amplitudine $A = 4$, frecvență $\omega = 0.1$ și fază $\varphi = 0$. Să se suprapună zgomot peste acest semnal, utilizând funcția Matlab `rand`. Implementați un filtru de mediere și observați efectele aplicării acestuia asupra semnalului afectat de zgomot, pentru M = 2...10 (vezi Figura 5.4).

5.4 Interpolatorul liniar

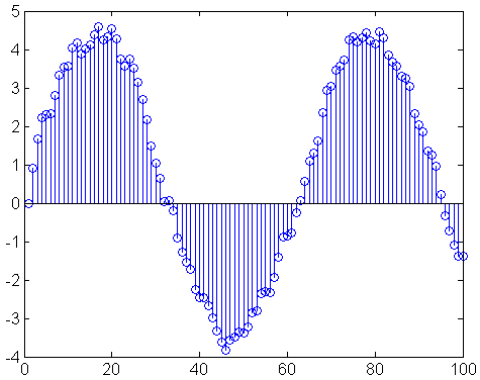
Interpolatorul liniar se utilizează pentru creșterea ratei de eșantionare a unui semnal discretizat în timp. Prima etapă o reprezintă supraeșantionarea secvenței de intrare $x[n]$ cu un factor L, prin introducerea de zerouri, rezultând o secvență $x_u[n]$ de $L \cdot N$ eșantioane.

$$x_u[n] = \begin{cases} x[(n-1)/L + 1], n = 1, L+1, 2L+1, ... \\ 0, \text{în rest} \end{cases} \tag{5.6}$$

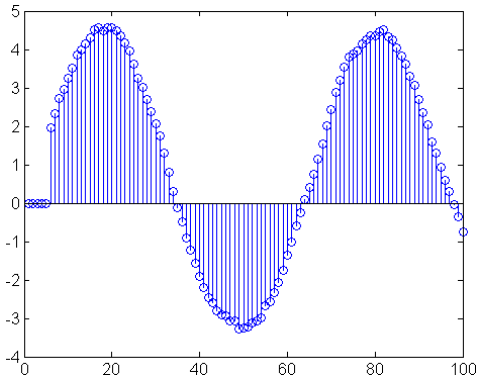


(a) Semnal sinusoidal.

(b) Semnalul afectat de zgomot.



(c) Rezultatul filtrării cu $M = 2$.



(d) Rezultatul filtrării cu $M = 6$.

Figura 5.4: Efectul aplicării unui filtru de mediere.

Interpolarea se aplică asupra semnalului supraeșantionat $x_u[n]$, pentru a estima valorile lipsă. Cel mai simplu mod de interpolare îl reprezintă interpolarea liniară. Relația pentru calculul semnalului de ieșire în cazul interpolării liniare cu un factor $L = 2$ este următoarea:

$$y[n] = x_u[n] + \frac{1}{2}\Big(x_u[n-1] + x_u[n+1]\Big), n = 2, ..., 2N-1 \tag{5.7}$$

În timp ce pentru o interpolare cu un factor $L = 3$, relația este următoarea:

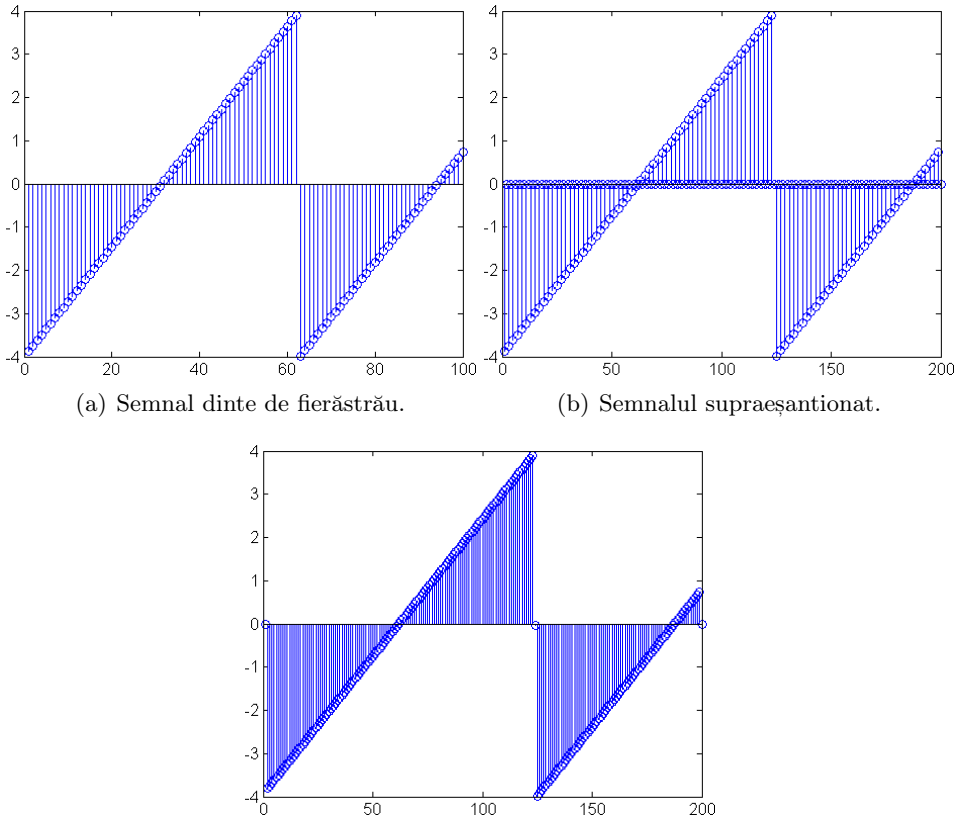
$$y[n] = x_u[n] + \frac{2}{3}\Big(x_u[n-1] + x_u[n+1]\Big) + \frac{1}{3}\Big(x_u[n-2] + x_u[n+2]\Big),$$
$$n = 3, ..., 3N-2 \tag{5.8}$$

Exerciții

1. Să se genereze $N = 100$ eșantioane ale unui semnal de tip dinte de fierăstrău de amplitudine $A = 4$, frecvență $\omega = 0.1$ și fază $\varphi = 0$. Dublați rata de eșantionare a acestui semnal, realizând o interpolare liniară cu un factor $L = 2$ (vezi Figura 5.5).

2. Triplați rata de eșantionare a aceluiași semnal, realizând o interpolare liniară cu un factor $L = 3$.

3. Deduceți și implementați ecuația unui interpolator liniar cu un factor L oarecare.



(a) Semnal dinte de fierăstrău.

(b) Semnalul supraeșantionat.

(c) Rezultatul interpolării liniare.

Figura 5.5: Interpolarea liniară cu factor $L = 2$.

Lucrarea 6

Filtrul trece-jos ideal

În această lucrare vom aborda problema filtrării liniare a semnalelor. Sistemele liniare invariante în timp (pe scurt, filtrele liniare) sunt de o importanță aparte în studiul prelucrării de semnale, în contextul analizei de tip Fourier a semnalelor. Ne propunem proiectarea și implementarea unui filtru trece-jos ideal, fără a ține cont de problema ferestruirii. Pentru o implementare reală, vom studia efectele aproximării funcției pondere teoretice a filtrului, analizând comportamentul în frecvență al filtrului (cu ajutorul transformatei Fourier) pentru câteva semnale sintetice.

6.1 Sisteme liniare invariante în timp

Un sistem se numește *liniar* dacă, aplicând la intrarea sa o combinație liniară a două semnale $x_1(t)$ și $x_2(t)$, de forma $ax_1(t) + bx_2(t)$ cu $a, b \in \mathbb{R}$, semnalul de ieșire va fi de forma $ay_1(t) + by_2(t)$, unde $y_1(t)$ și $y_2(t)$ sunt răspunsurile sistemului la intrările $x_1(t)$, respectiv $x_2(t)$ (vezi Figura 6.1).

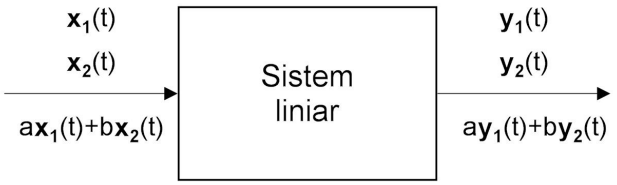


Figura 6.1: Sistem liniar.

Sistemul se numește *invariant în timp* dacă acțiunea sa asupra unui semnal de intrare $x(t)$ este aceeași indiferent de momentul la care este aplicat semnalul respectiv la intrarea sistemului (ilustrare în Figura 6.2). Cu alte cuvinte, pentru un același stimul (semnal) aplicat la intrare, răspunsul sistemului este același indiferent de momentul de timp la care este aplicat stimulul.

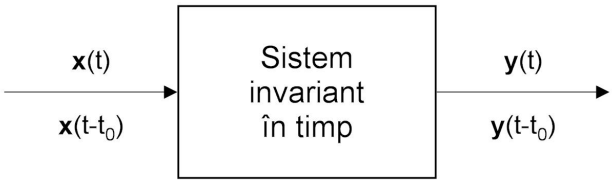


Figura 6.2: Sistem invariant în timp.

Sistemele liniare invariante în timp (pe care le vom numi, pe scurt, *filtre liniare*) au ca funcții proprii semnalele sinusoidale (de unde contextul analizei Fourier a semnalelor). Altfel spus, dacă aplicăm la intrarea unui filtru liniar un semnal sinusoidal de frecvență ω_0, semnalul de ieșire va fi tot o sinusoidă de frecvență ω_0 (cu amplitudinea și faza modificate față de sinusoida de intrare). Un filtru liniar va fi, deci, complet caracterizat de o funcție care descrie modificarea amplitudinii și a fazei sinusoidelor pentru fiecare frecvență ω_0, funcție care se numește *răspuns în frecvență* și se notează $H(\omega)$.

În cazul în care semnalul $x(t)$ aplicat la intrarea filtrului liniar nu este sinusoidal, putem explica forma semnalului de ieșire $y(t)$ tot prin reducere la cazul sinusoidal. Acest lucru este posibil întrucât, conform analizei Fourier a semnalelor, orice semnal de modul integrabil $x(t)$ poate fi descompus într-o sumă (infinită) de semnale pur sinusoidale.

Data fiind transformata Fourier a semnalului $x(t)$, anume $X(\omega)$, pentru *fiecare* componentă sinusoidală de frecvență ω din semnal, putem calcula faza și amplitudinea la ieșire în funcție de cele de la intrare ca și:

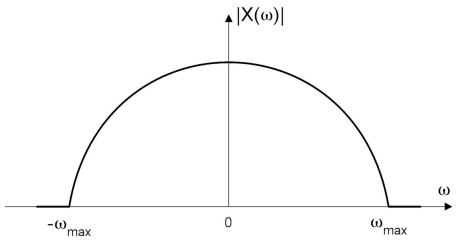
$$Y(\omega) = H(\omega)X(\omega) \quad \forall\omega, \tag{6.1}$$

după care recompunem semnalul de ieșire însumând toate sinusoidele $Y(\omega)$:

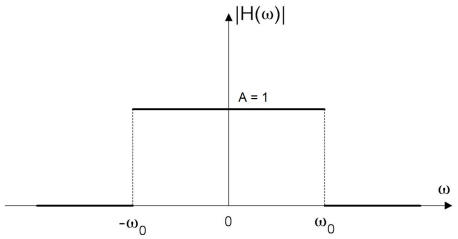
$$y(t) = \mathcal{F}^{-1}\{Y(\omega)\}(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} Y(\omega)e^{j\omega t}d\omega \tag{6.2}$$

6.2 Filtrul trece-jos ideal

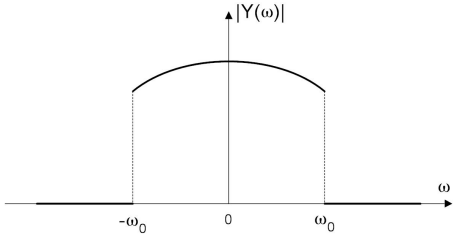
Filtrul trece-jos (FTJ) ideal, denumit și *filtrul Sinc*, este acel filtru care lasă să treacă nealterate toate componentele spectrale de frecvențe mai mici decât un anumit prag ω_0, în timp ce componentele de frecvențe superioare pragului respectiv sunt complet rejectate.



(a) Spectrul mărginit al semnalului de intrare.



(b) Modulul filtrului trece-jos ideal.



(c) Spectrul semnalului de la ieșirea filtrului.

Figura 6.3: Spectrele de amplitudine ale semnalului de intrare, răspunsului în frecvență al filtrului trece-jos și semnalului de ieșire.

6.2.1 Filtrarea trece-jos ideală în domeniul frecvență

Fie un semnal $x(t)$ având un spectru mărginit, de forma celui din Figura 6.3(a). Ne propunem să filtrăm acest semnal cu un filtru trece-jos ideal ce are frecvența de tăiere $\omega_0 < \omega_{max}$. Modulul răspunsului în frecvență al unui astfel de filtru este prezentat în Figura 6.3(b). Rezultatul operației de filtrare în domeniul frecvență va fi $Y(\omega) = H(\omega)X(\omega)$; deci, la ieșirea filtrului, semnalul $y(t)$ va avea, teoretic, un spectru Fourier de amplitudine de forma celui din Figura 6.3(c). Bineînțeles, semnalul $y(t)$ poate fi reconstituit, prin transformarea Fourier inversă, din spectrul $Y(\omega)$, conform relației (6.2). Etapele descrise mai sus sunt prezentate într-o forma schematizată în Figura 6.4.

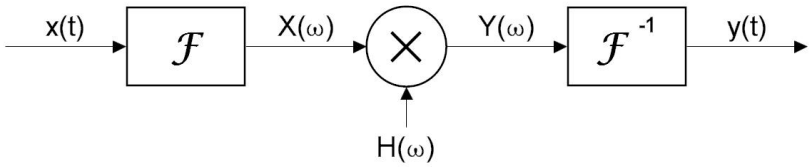


Figura 6.4: Schema bloc a filtrării în domeniul frecvență.

Comportamentul filtrului este modelat cu ajutorul unei funcții de tip răspuns în frecvență de forma $H(\omega) = |H(\omega)|e^{j\phi(\omega)}$. În general, un număr complex c sau o funcție complexă se poate scrie sub formă exponențială în funcție de modul și fază, astfel :

$$c = |c|e^{j\phi} \tag{6.3}$$

unde $|c|$ reprezintă modulul, iar ϕ faza.

În cazul răspunsului în frecvență al filtrului trece-jos ideal, modulul filtrului (vezi Figura 6.3(b)) este dat de:

$$|H(\omega)| = \begin{cases} A & \text{dacă } |\omega| \leq \omega_0 \\ 0 & \text{în rest} \end{cases} \tag{6.4}$$

iar faza filtrului este dată de:

$$\phi(\omega) = \begin{cases} -\omega t_0 & \text{dacă } |\omega| \leq \omega_0 \\ 0 & \text{în rest} \end{cases} \tag{6.5}$$

Răspunsul în frecvență al filtrului devine astfel:

$$H(\omega) = \begin{cases} Ae^{-j\omega t_0} & \text{dacă } |\omega| \leq \omega_0 \\ 0 & \text{în rest} \end{cases} \tag{6.6}$$

Frecvența ω_0 se numește frecvență de tăiere a filtrului și reprezintă frecvența maximă din semnalul de intrare care este lăsată să treacă de filtru, iar t_0 se numește timp de întârziere de grup și reprezintă întârzierea indusă de filtru în semnal. Intervalul de frecvențe ω care respectă condiția $|\omega| \leq \omega_0$ se numește bandă de trecere a filtrului. Frecvențele care nu respectă condiția de mai sus alcătuiesc banda de oprire.

Pentru un filtru trece-jos care nici nu amplifică, nici nu atenuează componentele de frecvență cuprinse în banda de trecere, $A = 1$.

6.2.2 Filtrarea trece-jos ideală în domeniul timp

Considerăm transformata inversă a răspunsului în frecvență, notată $h(t)$ și numită *funcție pondere*:

$$h(t) = \mathcal{F}^{-1}\{H(\omega)\}(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} X(\omega)e^{j\omega t}d\omega \tag{6.7}$$

Astfel, ținând cont de proprietatea transformatei Fourier privind produsul de convoluție, relația (6.1) poate fi scrisă direct în domeniul temporal:

$$y(t) = h(t) \star x(t) = \int_{-\infty}^{+\infty} h(\tau)x(t-\tau)d\tau = \int_{-\infty}^{+\infty} h(t-\tau)x(\tau)d\tau \tag{6.8}$$

Comportamentul filtrului poate fi descris astfel în domeniul timp, utilizând funcția pondere $h(t)$ în locul răspunsului în frecvență (vezi Figura 6.5).

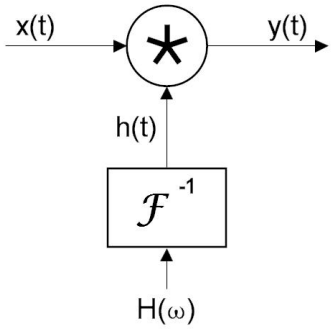


Figura 6.5: Schema bloc a filtrării în domeniul timp.

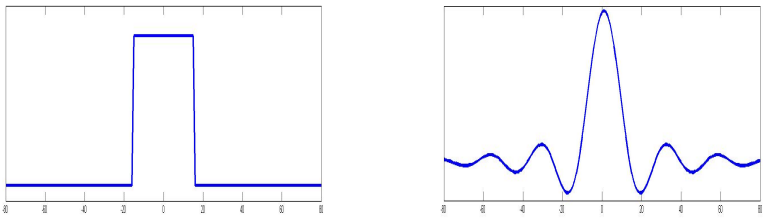
Funcția pondere $h(t)$ a filtrului trece-jos ideal se obține aplicând transformata Fourier inversă răspunsului în frecvență:

$$\begin{aligned} h(t) =& \mathcal{F}^{-1}\{H(\omega)\}(t) = \frac{1}{2\pi}\int_{-\omega_0}^{+\omega_0} Ae^{-j\omega t_0}e^{j\omega t}d\omega \\ =& \frac{A}{2\pi}\frac{1}{j(t-t_0)}e^{j\omega(t-t_0)}\Big|_{-\omega_0}^{+\omega_0} = \frac{A}{2\pi}\frac{1}{j(t-t_0)}2j\sin(\omega_0(t-t_0)) \\ =& \frac{A\omega_0}{\pi}\text{sinc}(\omega_0(t-t_0)) \end{aligned} \tag{6.9}$$

unde am folosit relația $\sin(x) = \frac{e^{jx}-e^{-jx}}{2j}$, iar $\text{sinc}(x)$ este funcția sinus cardinal, definită ca:

$$\text{sinc}(x) \triangleq \frac{\sin(x)}{x} \tag{6.10}$$

Relația bijectivă dintre $H(\omega)$ și $h(t)$ este ilustrată în Figura 6.6.



(a) $H(\omega) = \mathcal{F}\{h(t)\}$ (b) $h(t) = \mathcal{F}^{-1}\{H(\omega)\}$

Figura 6.6: Relația de bijectivitate dintre răspunsul în frecvență (a) și funcția pondere (b) ale unui FTJ prin transformata Fourier.

Deoarece un FTJ nu trebuie să altereze componenta continuă a unui semnal, cu alte cuvinte componenta de frecvență $\omega = 0$ trebuie să treacă prin filtru, pentru un semnal de intrare constant $x(t) = K$, ieșirea filtrului va fi:

$$y(t) = h(t) \star x(t) = \int_{-\infty}^{+\infty} h(\tau)x(t-\tau)d\tau = K \int_{-\infty}^{+\infty} h(\tau)d\tau \tag{6.11}$$

Cum ieșirea filtrului pentru semnalul constant $x(t)$ va fi $y(t) = AK$, rezultă că funcția pondere a filtrului trece-jos trebuie să respecte următoarea condiție de normare:

$$\int_{-\infty}^{+\infty} h(t)dt = A \tag{6.12}$$

Se observă că $h(t) \neq 0$ pentru $t < 0$, de unde rezultă că filtrul nu este cauzal și, deci, nu poate fi implementat într-un sistem real. Un filtru se numește cauzal dacă semnalul de ieșire apare ca rezultat al aplicării unui semnal la intrarea filtrului, aceasta fiind echivalentă cu condiția $h(t) = 0, \forall t < 0$.

6.3 Desfășurarea lucrării

În cadrul acestei lucrări se va genera un semnal alcătuit dintr-o sumă de sinusoide, se va realiza filtrarea trece-jos a semnalului în domeniul frecvență, respectiv în domeniul timp și se vor analiza rezultatele obținute.

6.3.1 Filtrarea trece-jos în domeniul frecvență

Să se genereze un semnal alcătuit din patru sinusoide, de forma:

$$x(t) = A_1 \sin(\omega_1 t) + A_2 \sin(\omega_2 t + \varphi_2) + A_3 \sin(\omega_3 t) + A_4 \sin(\omega_4 t + \varphi_4) \quad (6.13)$$

cu $A_1 = 0.25$, $\omega_1 = \omega = 0.1$, $A_2 = 0.5$, $\omega_2 = 2\omega$, $\varphi_2 = 10$, $A_3 = 1$, $\omega_3 = 4\omega$ și $A_4 = 1.5$, $\omega_4 = 5\omega$, $\varphi_4 = 25$.

Secvența Matlab pentru generarea și afișarea a N=512 valori ale acestui semnal este:

```
N = 512;
omega = 0.1;
t = 1 : N;
x = 0.25*sin(omega*t) + 0.5*sin(2*omega*t+10)...
    + 1*sin(4*omega*t) + 1.5*sin(5*omega*t+25);
figure;
plot(t, x);
```

Forma semnalului obținut va fi cea din Figura 6.7(a).

Să se realizeze filtrarea în domeniul frecvență a semnalului generat cu un FTJ având frecvența de tăiere $\omega_0 \approx 2\omega$.

Primul pas constă în obținerea spectrului $X(\omega)$ din semnalul $x(t)$ prin transformata Fourier:

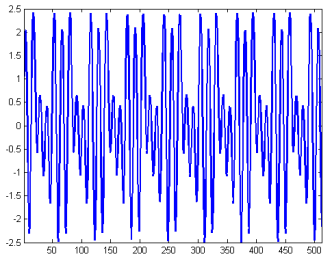
```
X = fftshift(fft(x));
figure;
subplot(3,1,1),bar(abs(X));
```

Se observă în spectrul de amplitudine al semnalului (Figura 6.8 sus), în partea dreaptă față de mijlocul axei Ox, componentele de frecvențe ω, 2ω, 4ω, 5ω și, în oglindă, cele de frecvențe $-\omega$, -2ω, -4ω, -5ω.

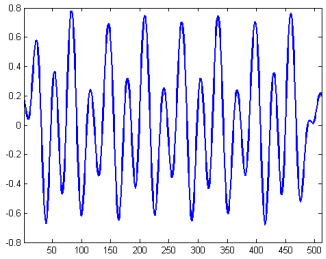
Următorul pas constă în generarea a N valori ale răspunsului în frecvență $H(\omega)$, astfel încât în urma operației de filtrare să se păstreze doar componentele de frecvențe $\pm\omega$ și $\pm 2\omega$ (vezi Figura 6.8 mijloc). Valoarea funcției pentru banda de trecere va fi 1, iar în rest 0:

```
H = zeros(1,N);
P = 20;
H(N/2 + 1 - P : N/2 + 1 + P) = 1;
subplot(3,1,2),bar(abs(H));
```

Se realizează filtrarea, în urma căreia se obține spectrul semnalului de ieșire $Y(\omega) = H(\omega)X(\omega)$ (vezi Figura 6.8 jos) :



(a) $x(t)$



(b) $y(t)$

Figura 6.7: Semnalele de la intrarea (a) și ieșirea (b) filtrului trece-jos proiectat.

```
Y = H.*X;
subplot(3,1,3),bar(abs(Y));
ylim([0 max(abs(X))]);
```

Se observă că spectrul obținut conține doar componentele dorite, de frecvențe $\pm\omega$ și $\pm 2\omega$, frecvențele mai mari decât ω_0 fiind complet eliminate.

Ultimul pas îl reprezintă obținerea și afișarea semnalului $y(t)$ de la ieșirea filtrului prin transformata Fourier inversă (vezi Figura 6.7(b)) :

```
y = ifft(fftshift(Y));
figure;
plot(t, y);
```

6.3.2 Filtrarea trece-jos în domeniul timp

Ne propunem să implementăm operația de filtrare din secțiunea precedentă, de data aceasta în domeniul timp, prin obținerea și trunchierea funcției pondere și realizarea operației de convoluție între eșantioanele funcției pondere și semnalul de intrare.

Se consideră cele N valori ale semnalului $x(t)$ generate la punctul anterior. Răspunsul în frecvență va fi de asemenea identic cu cel precedent, generat astfel încât frecvența de tăiere a filtrului să fie $\omega_0 \approx 2\omega$:

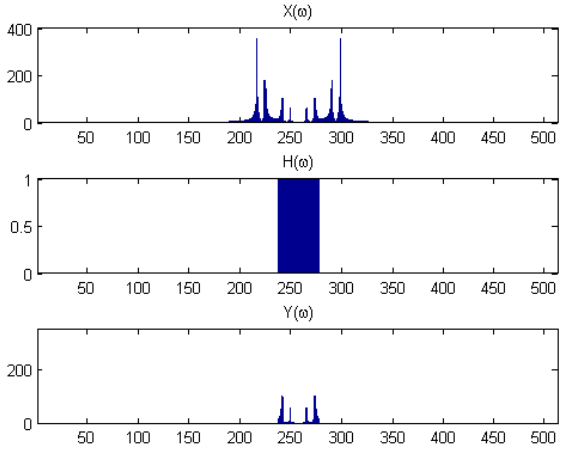


Figura 6.8: Spectrele $X(\omega)$, $H(\omega)$ și $Y(\omega)$ în cazul filtrării în domeniul frecvență.

```
H = zeros(1,N);
P = 20;
H(N/2 + 1 - P : N/2 + 1 + P) = 1;
```

Funcția pondere $h(t)$ se obține prin aplicarea transformatei Fourier inverse asupra răspunsului în frecvență, păstrându-se doar partea reală a valorilor obținute:

```
h = fftshift(ifft(fftshift(H)));
figure;
plot(t, h);
```

Se trunchiază funcția pondere de tip *sinc*, preluându-se $2L+1$ eșantioane consecutive din $h(t)$ în jurul lobului central (vezi Figura 6.9), obținându-se vectorul h_es. Pentru respectarea condiției de normare (6.12), valorile obținute vor fi ponderate cu inversul sumei tuturor valorilor din vectorul generat. Secvența pentru generarea h_es cu 15 valori este:

```
L = 7;
t_es = (N/2 + 1 - L) : (N/2 + 1 + L);
h_es = h(t_es);
hold on;
stem(t_es, h_es);
hold off;
h_es = (1/sum(h_es)).*h_es;
```

Semnalul de ieșire $y(t)$ se obține prin convoluția dintre eșantioanele funcției pondere și semnalul de intrare, realizată utilizând funcția Matlab `filter`:

```
y = filter(h_es,1,x);
figure;
plot(t, y);
```

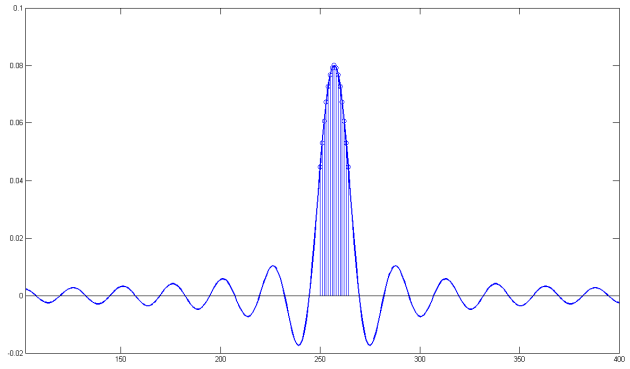


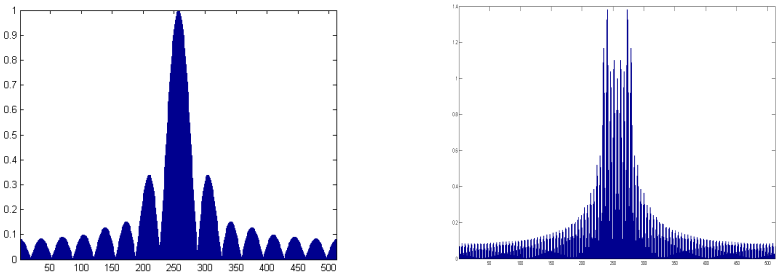
Figura 6.9: Funcția pondere și valorile obținute în urma trunchierii.

Se afișează spectrele de amplitudine ale semnalului de intrare, răspunsului în frecvență rezultat în urma trunchierii funcției pondere și semnalului de ieșire:

```
X = fftshift(fft(x));
figure;
subplot(3,1,1),bar(abs(X));

H_es = fftshift(fft(h_es,N));
subplot(3,1,2),bar(abs(H_es));

Y = fftshift(fft(y));
subplot(3,1,3),bar(abs(Y));
ylim([0 max(abs(X))]);
```

Care sunt diferențele față de spectrul obținut la filtrarea în domeniul frecvență pentru semnalul de ieșire?

Repetați pașii operației de filtrare în domeniul timp pentru 31, respectiv 101 de eșantioane ale funcției pondere. Ce observați?



(a) (b)

Figura 6.10: Modulul răspunsului în frecvență rezultat în urma preluării a 15 eșantioane (a), respectiv 101 eșantioane (b) din funcția pondere.

6.4 Exerciții

1. Implementați o funcție Matlab `myfilter`, care întoarce un vector c ca rezultat al convoluției dintre doi vectori a și b, realizată conform formulei:

$$c(i) = \sum_{k=1}^{A} a(k)b(i-k), \quad \text{pentru } i = A+1, ..., B$$

   unde A și B reprezintă numărul de valori ale vectorilor a, respectiv b.

2. Repetați pașii operației de filtrare în domeniul frecvență pentru un semnal de intrare de tip impuls Dirac, respectiv un semnal de tip dinte de fierăstrău de frecvență $\omega = 0.05$.

Lucrarea 7

Modelarea semnalelor ca procese aleatoare

Scopul lucrării este acela de a familiariza studenţii cu generarea semnalelor aleatoare cu parametri cunoscuţi şi cu o distribuţie dată, determinarea funcţiei de repartiţie şi a densităţii de probabilitate, calcularea mediei şi a varianţei.

7.1 Caracterizarea unui proces aleator

Un semnal aleator este un proces care se desfăşoară în timp şi este guvernat de legi probabilistice. Din punct de vedere matematic, un semnal aleator este o funcţie de două variabile $\xi(k,t) = \xi^{(k)}(t)$, unde k ia valori în spaţiul eşantioanelor, iar t ia valori pe axa reală a timpului. Funcţia $\xi^{(k)}(t)$ face parte din mulţimea sau clasa de semnale $\xi(t)$ şi se numeşte o „realizare particulară" a procesului $\xi(t)$.

Pentru a caracteriza un semnal aleator la un moment de timp t arbitrar sau pentru o realizare particulară, se folosesc funcţiile de repartiţie şi cea de densitate de probabilitate. Alte mărimi caracteristice larg utilizate sunt momentele statistice, cele mai importante fiind media şi varianţa.

Funcţia de repartiţie, definită într-un punct x, este probabilitatea ca variabila aleatoare la momentul t să ia valori mai mici sau egale decât pragul x:

$$F_\xi(x,t) = P\{\xi(t) \leq x\}$$

Densitatea de probabilitate este derivata funcţiei de repartiţie, şi anume:

$$w_\xi(x,t) = \frac{dF_\xi(x,t)}{dx}$$

7.1.1 Momente statistice

Există o serie de momente statistice utilizate în diverse aplicații, cele mai importante fiind:

1. Valoarea medie
$$\overline{\xi(t)} = \int_{-\infty}^{+\infty} x w_\xi(x,t) dx$$

2. Valoarea pătratică medie
$$\overline{\xi^2(t)} = \int_{-\infty}^{+\infty} x^2 w_\xi(x,t) dx$$

3. Varianţa
$$\sigma^2(t) = \overline{\xi^2(t)} - \overline{\xi(t)}^2$$

7.2 Generarea proceselor aleatoare

Se vor genera trei tipuri de semnale aleatoare:

- $u(n)$ cu distribuţie uniformă
- $g(n)$ cu distribuţie Gaussiană (normală)
- $r(n)$ cu distribuţie Rayleigh

Cele trei semnale vor fi generate folosind următoarele formule:

$$u(n) = m + \sigma\sqrt{3}(2\xi - 1) \tag{7.1}$$

$$g(n) = m + \sigma\sqrt{-2ln\xi}cos(2\pi\eta) \tag{7.2}$$

$$r(n) = m + \sigma\sqrt{-2ln\xi} \tag{7.3}$$

unde funcţia ξ reprezintă o variabilă aleatoare distribuită uniform în intervalul $[0, 1]$, obţinută cu ajutorul funcţiei Matlab `rand`. Acelaşi lucru este valabil şi pentru variabila η. Secvența Matlab pentru generarea a câte $N = 1000$ de eșantioane pentru fiecare din cele trei semnale este următoarea:

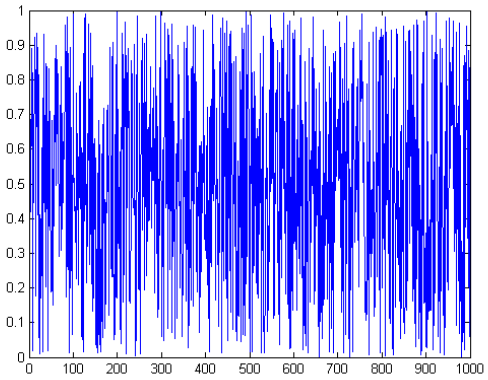


Figura 7.1: Reprezentarea grafică a valorilor lui ξ.

```
m = 3;
sigma = 2;
N = 1000;
eta = rand(1,N);
xi = rand(1,N);
u = m + sigma*sqrt(3)*(2*xi-1);
g = m + sigma*sqrt(-2*log(xi)).*cos(2*pi*eta);
r = m + sigma*sqrt(-2*log(xi));
```

Se vor vizualiza diferite realizări particulare ale celor 3 semnale, folosind funcțiile `figure` şi `plot` din Matlab, pentru mai multe valori ale mediei şi ale varianţei.

Cu ajutorul funcţiilor `mean` şi `var` din Matlab se vor calcula mediile şi varianţele celor 3 semnale şi se vor compara cu mediile şi varianţele specificate. Cum explicaţi diferenţele?

7.3 Estimarea funcției de repartiție și a densității de probabilitate

Se va determina funcţia de repartiţie pentru fiecare din cele 3 realizări particulare ale semnalelor aleatoare, pe baza algoritmului următor:

- Se alege un număr L de nivele de cuantizare (de exemplu $L = 100$).
- În funcţie de valorile minime şi maxime ale semnalelor, determinate cu funcţiile Matlab `min`, respectiv `max`, se va calcula pasul de cuantizare:

$$\delta = \frac{max - min}{L}$$

- Se va genera un vector de L valori discrete ale variabilei x în intervalul $[min, max]$, cu pasul δ, utilizând formula:

$$x[j] = min + j\delta$$

pentru $j = 1, ..., L$.

- Se vor calcula L valori ale funcției de repartiție a semnalului, pe baza formulei:

$$F_\xi[j] = P\{\xi \leq x[j]\}$$

pentru $j = 1, ..., L$. Probabilitatea se va calcula ca frecvenţă relativă de apariţie, cu alte cuvinte ca raport între numărul de valori mai mici decât un anume $x[j]$ şi numărul total de valori ale semnalului.

Se va reprezenta grafic funcţia de repartiţie.

Pornind de la definiţie, se va determina funcţia de densitate de probabilitate, ca fiind derivata funcției de repartiție :

$$w_\xi[j] = \frac{dF_\xi[j]}{dx[j]}$$

Pentru cazul discret, operația de derivare va fi implementată utilizând formula:

$$w_\xi[j] = \frac{F_\xi[j] - F_\xi[j-1]}{x[j] - x[j-1]} = \frac{F_\xi[j] - F_\xi[j-1]}{\delta}$$

pentru $j = 2, ..., L$.

Reprezentaţi grafic funcţia de densitate de probabilitate astfel calculată. Rulați algoritmul de mai multe ori, crescând numărul de eșantioane ale semnalului generat (de exemplu $N = 5000, 10000, ...$).

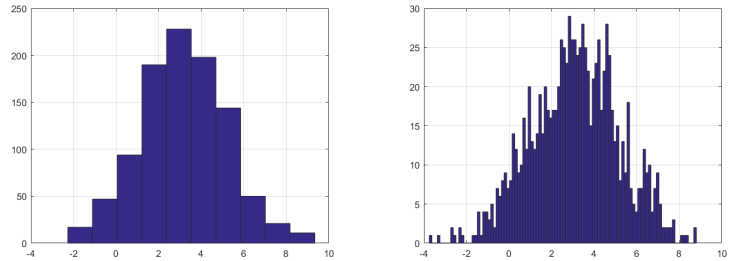
Să se calculeze histograma valorilor semnalelor, folosind funcţia Matlab `hist` şi să se reprezinte grafic. Histograma reprezintă un estimat al funcţiei de densitate de probabilitate. Comparaţi rezultatul obţinut cu graficul anterior. Ce observaţi?

În Figura 7.2 sunt prezentate câteva exemple de histograme pentru variabile aleatoare distribuite normal, pentru diverse valori ale parametrilor N și L. Cum explicați diferențele?

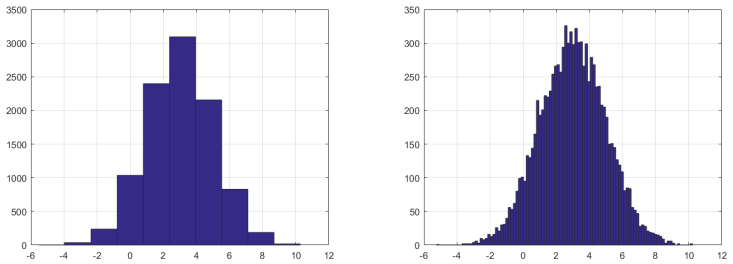
Histograma cumulativă $h_c(x)$ reprezintă estimatul funcţiei de repartiţie şi se calculează pe baza histogramei $h(x)$ folosind formula:

$$h_c(x) = \int_{-\infty}^{x} h(\tau)d\tau$$

În cazul discret, integrala se transformă într-o sumă, iar formula de calcul a histogramei cumulative devine:



(a) $N = 1000$, $L = 10$ (b) $N = 1000$, $L = 100$



(c) $N = 10000$, $L = 10$ (d) $N = 10000$, $L = 100$

Figura 7.2: Exemple de histograme pentru variabile aleatoare distribuite normal.

$$h_x(x) = \sum_{k=-\infty}^{x} h(k)$$

Calculaţi şi reprezentaţi grafic histograma cumulativă. Ce observaţi?

7.4 Exerciții

1. Determinaţi şi reprezentaţi grafic histograma pentru următoarea secvenţă de valori: 1 1 3 2 3 2 4 4 5 4 1 1 1 4 3 4 2 4 5 4 4 4 3 5 5 3 3.

2. Calculaţi şi reprezentaţi grafic histograma cumulativă.

Lucrarea 8

Cuantizarea semnalelor

Cuantizarea reprezintă a doua operație realizată în cadrul conversiei unui semnal din domeniul continuu în cel discret, după operația de eșantionare și anume discretizarea în valoare a semnalului. Scopul acestei lucrări este de a prezenta două tipuri de cuantizare: cuantizarea uniformă, respectiv cuantizarea Lloyd–Max.

8.1 Funcția de cuantizare

Operația de cuantizare realizează discretizarea *în amplitudine* sau în valoare a eșantioanelor, prin aproximarea valorilor continue ale acestora cu valori fixe, provenite dintr-o mulțime finită. Operația premergătoare de eșantionare realizează o discretizare *în timp* a semnalului. O observație importantă este aceea că operația nu este bijectivă; cu alte cuvinte, nu se mai poate reface forma originală a semnalului după ce acesta a fost cuantizat. Prin urmare, cuantizarea este o operație însoțită de zgomot, numit *zgomot de cuantizare*.

Considerând că gama de valori a semnalului discret de intrare $\xi[n]$ este limitată în intervalul $[\xi_{\min},\xi_{\max}]$, se poate defini o funcție de cuantizare Q. Aceasta modelează o cuantizare pe L nivele, asociind fiecărui argument x o valoare y dintr-o mulțime finită cu L elemente, după cum urmează:

$$y = Q(x) = \begin{cases} y_L & \text{dacă } x_L < x \leq x_{L+1} \\ y_{L-1} & \text{dacă } x_{L-1} < x \leq x_L \\ \cdots & \\ y_1 & \text{dacă } x_1 \leq x \leq x_2 \end{cases} \tag{8.1}$$

unde $x_1 = \xi_{\min}$ iar $x_{L+1} = \xi_{\max}$.

Astfel, semnalul de ieșire va fi:

$$\eta[n] = Q(\xi[n]) \tag{8.2}$$

Valorile x_i se numesc *praguri de cuantizare*, în timp ce valorile y_i se numesc *valori cuantizate*. Diferența între două praguri de cuantizare succesive x_i și x_{i+1} se notează cu Δ_i și este denumită *pas de cuantizare* sau *cuantă*. Specificarea unei funcții de cuantizare constă în alegerea numărului L de nivele ale cuantizorului, iar apoi în definirea pragurilor de cuantizare $x_2, \ldots, x_L$ și a valorilor cuantizate $y_1, \ldots, y_L$.

Forma generală a graficului unei funcții de cuantizare este prezentată în Figura 8.1, în timp ce Figura 8.2 ilustrează forma unui semnal înainte și după cuantizare. Pentru conversia digitală, valorile rezultate în urma cuantizării sunt codate pe un număr de N biți, astfel încât $2^N \geq L$.

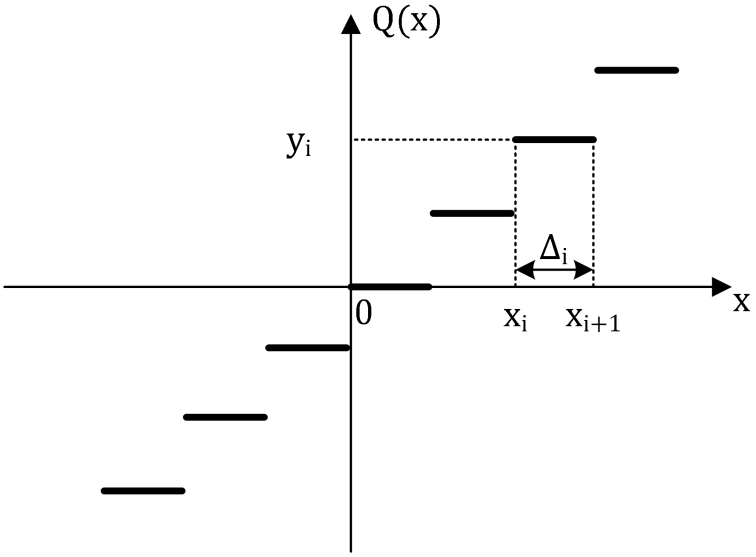


Figura 8.1: Forma generală a graficului unei funcții de cuantizare.

8.2 Cuantizarea uniformă

Cel mai des întâlnit tip de cuantizare este cea uniformă, realizată prin alegerea unui pas de cuantizare unic Δ pe întreg intervalul de valori ale semnalului de intrare:

$$\Delta_{i-1} = \Delta_i = \Delta \quad \forall i = 2, 3, \ldots, L \tag{8.3}$$

Prin urmare, valoarea pasului de cuantizare va fi:

$$\Delta = \frac{\xi_{\max} - \xi_{\min}}{L} \tag{8.4}$$

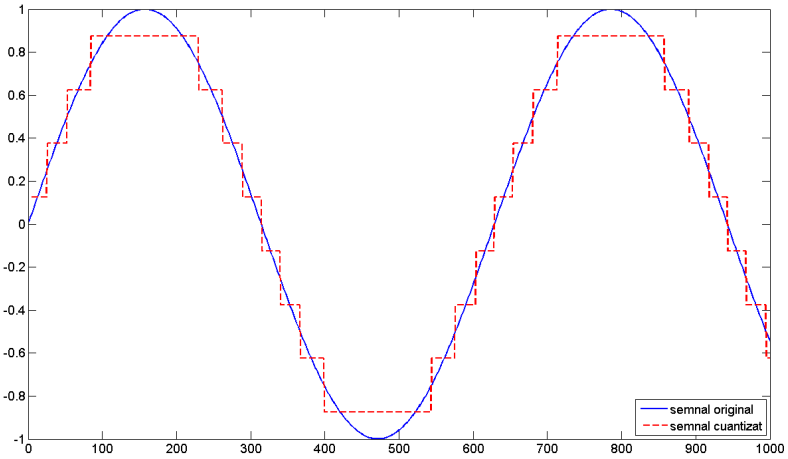


Figura 8.2: Forma unui semnal sinusoidal înainte și după cuantizare.

Următorul pas este calculul valorilor pragurilor de cuantizare $x_2, \ldots, x_L$ ($x_1 = \xi_{\min}$ și $x_{L+1} = \xi_{\max}$ sunt deja cunoscute):

$$x_i = \xi_{\min} + (i-1)\Delta, \quad \forall i = 2, 3, \ldots, L \tag{8.5}$$

În final, valorile cuantizate se aleg la mijlocul fiecărui interval de cuantizare:

$$y_i = \frac{x_i + x_{i+1}}{2} \quad \forall i = 1, 2, \ldots, L \tag{8.6}$$

Alegerea valorilor cuantizate reprezintă o convenție, la fel de bine putând fi alese și capetele intervalelor de cuantizare.

Diferența dintre semnalul de intrare și cel cuantizat se numește *eroare de cuantizare* sau *zgomot de cuantizare* și se poate exprima ca o funcție g aplicată semnalului de intrare:

$$e[n] = \xi[n] - \eta[n] = g(\xi[n]) \tag{8.7}$$

Graficul funcției g corespunzătoare unei cuantizări uniforme, dată de funcția Q din Figura 8.1, este prezentat în Figura 8.3.

8.3 Cuantizarea optimală Lloyd–Max

Cuantizarea uniformă nu este una optimală, deoarece nu ține cont de distribuția valorilor semnalului de intrare. Ca urmare, zgomotul de cuantizare nu este minimizat - puterea medie a acestuia ar putea fi minimizată prin specificarea unei funcții Q pe baza funcției de densitate de

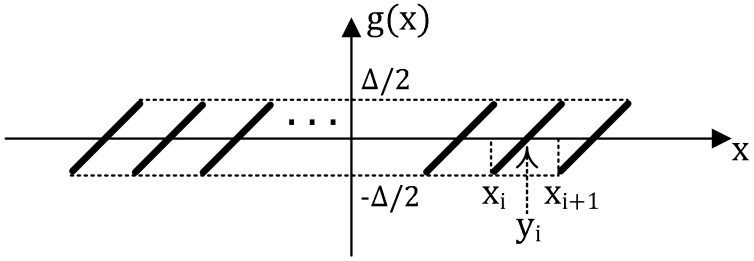


Figura 8.3: Funcția de eroare pentru cuantizarea uniformă.

probabilitate a valorilor eșantioanelor semnalului de cuantizat. Astfel, o variantă îmbunătățită o constituie cuantizarea Lloyd–Max, care realizează alocarea unor pași de cuantizare mici pe intervalele de valori ale semnalului cu probabilitate mare de apariție, respectiv pași de cuantizare mai mari (și deci erori mai mari) pentru valorile mai puțin probabile. Definirea unei reguli de cuantizare Lloyd–Max constă în calculul pragurilor de cuantizare $x_2, \ldots, x_L$ și al valorilor cuantizate $y_1, \ldots, y_L$ astfel încât eroarea pătratică medie indusă de cuantizare să fie minimă.

Eroarea pătratică medie ε este dată de relația:

$$\varepsilon = \overline{(\xi - \eta)^2} = \int_{x_1}^{x_{L+1}} [x - Q(x)]^2 w_\xi(x) dx \tag{8.8}$$

unde semnalul de ieșire $\eta = Q(x)$, L este numărul de nivele de cuantizare, iar $w_\xi(x)$ este funcția de densitate de probabilitate a valorilor eșantioanelor semnalului de intrare.

Putem rescrie formula erorii pătratice medii, înlocuind $Q(x)$ cu valorile cuantizate corespunzătoare, y_i:

$$\varepsilon = \sum_{i=1}^{L} \int_{x_i}^{x_{i+1}} (x - y_i)^2 w_\xi(x) dx \tag{8.9}$$

În continuare, se pune problema minimizării acestei erori în raport cu valorile pragurilor x_i și valorile cuantizate y_i; prin urmare, vom impune ca derivatele parțiale ale erorii pătratice medii să fie nule:

$$\begin{cases} \frac{\partial \varepsilon}{\partial x_i} = 0 & \forall i = 2, 3, \ldots, L \\ \frac{\partial \varepsilon}{\partial y_i} = 0 & \forall i = 1, 2, \ldots, L \end{cases} \tag{8.10}$$

Pentru valorile pragurilor, ecuația devine:

$$\frac{\partial \varepsilon}{\partial x_i} = 2(x_i - y_{i-1})^2 w_\xi(x_i) - 2(x_i - y_i)^2 w_\xi(x_i) = 0 \tag{8.11}$$

care este echivalentă cu:

$$(x_i - y_{i-1})^2 = (x_i - y_i)^2 \implies -2x_i y_{i-1} + 2x_i y_i = y_i^2 - y_{i-1}^2 \tag{8.12}$$

De unde obținem:

$$x_i = \frac{y_{i-1} + y_i}{2} \quad \forall i = 2, \ldots, L \tag{8.13}$$

Cât despre valorile cuantizate y_i, ele se obțin din ecuația:

$$\frac{\partial \varepsilon}{\partial y_i} = -2 \int_{x_i}^{x_{i+1}} (x - y_i) w_\xi(x) dx = 0 \tag{8.14}$$

Valorile cuantizate finale vor fi:

$$y_i = \frac{\int_{x_i}^{x_{i+1}} x w_\xi(x) dx}{\int_{x_i}^{x_{i+1}} w_\xi(x) dx} \quad \forall i = 1, 2, \ldots, L \tag{8.15}$$

Astfel, din ecuația (8.13) putem obține valorile pragurilor x_i în funcție de valorile cuantizate y_{i-1} și y_i, în timp ce ecuația (8.15) permite calculul valorilor cuantizate y_i în funcție de valorile pragurilor x_i și x_{i+1}. Va rezulta un sistem de $2L - 1$ ecuații cu $2L - 1$ necunoscute, a cărui rezolvare duce la definirea valorilor pragurilor și a valorilor cuantizate ale unui cuantizor optimal Lloyd–Max cu L nivele de cuantizare.

8.4 Desfășurarea lucrării

În cadrul acestei lucrări se va realiza cuantizarea uniformă a unui semnal sinusoidal, apoi cuantizarea uniformă, respectiv optimală a unui semnal aleator cu distribuție normală și se vor analiza rezultatele obținute.

8.4.1 Cuantizarea uniformă

Să se genereze semnalul de intrare ξ, sub forma unei sinusoide de frecvență $\omega = 0.01$. Secvența Matlab pentru generarea și afișarea unui astfel de semnal este:

```
N = 1000;
t = 1 : N;
omega = 0.01;
xi = sin(omega*t);
figure;
plot(xi);
```

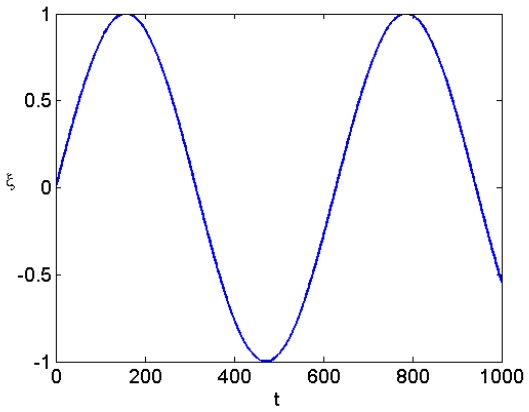


Figura 8.4: Semnalul sinusoidal de intrare pentru cuantizarea uniformă.

Forma semnalului generat este prezentată în Figura 8.4.

Să se realizeze cuantizarea uniformă a acestui semnal pe un număr de $L = 8$ nivele. Pentru realizarea operației este necesar calculul pasului de cuantizare Δ, pragurilor de cuantizare x și valorilor cuantizate y, conform formulelor (8.4), (8.5) și (8.6):

```
L = 8;
Delta = (max(xi) - min (xi))/ L;

x(1) = min(xi);
x(L+1) = max(xi);

i = 2:L;
x(2:L) = min(xi) + (i-1) * Delta;

for i=1:L
    y(i) = (x(i) + x(i+1))/2;
end
```

După definirea parametrilor se realizează operația efectivă, prin parcurgerea întregului vector de intrare, compararea fiecărei valori a acestuia cu pragurile de cuantizare x și atribuirea în vectorul de ieșire η a valorii cuantizate corespunzătoare din vectorul y. Se va afișa apoi forma semnalului de ieșire, respectiv graficul semnalului de ieșire în funcție de semnalul de intrare (o aproximare a graficului funcției de cuantizare).

```
for j = 1 : N
    for i = 1 : L
```

```
        if xi(j) >= x(i) && xi(j) <= x(i+1)
            eta(j) = y(i);
        end
    end
end

figure;
plot(eta);

figure;
plot (xi, eta, '*');
```

Forma semnalului de ieșire este prezentată în Figura 8.5, în timp ce graficul ieșirii în funcție de intrare se regăsește în Figura 8.6. Se pot observa pe acest al doilea grafic palierele de aceeași lățime, spațiate egal, specifice unei funcții de cuantizare uniformă.

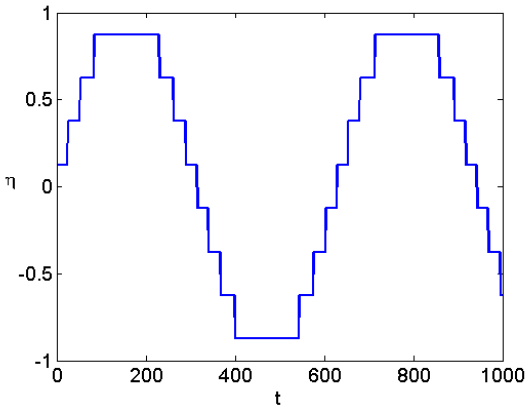


Figura 8.5: Semnalul sinusoidal cuantizat.

8.4.2 Cuantizarea optimală Lloyd–Max

Se va genera și afișa un semnal aleator cu distribuție normală, de forma celui din Figura 8.7(a):

```
N = 1000;
xi = randn (1, N);

figure;
plot(xi);
```

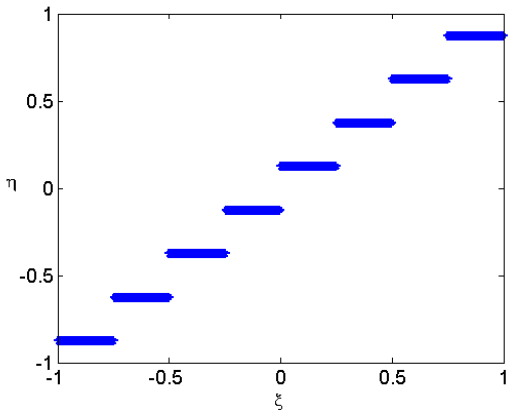
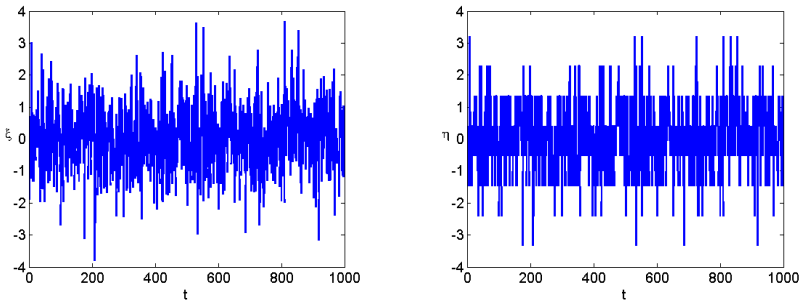


Figura 8.6: Funcția de cuantizare uniformă pentru semnalul sinusoidal.

Mai intâi se va realiza cuantizarea uniformă pe 8 nivele a acestui semnal, prin repetarea pașilor prezentați în secțiunea precedentă. Forma semnalului de ieșire va fi de genul celei din Figura 8.7(b).



(a) (b)

Figura 8.7: Semnalul aleator cu distribuție normală : (a) original; (b) cuantizat uniform.

Graficul ieșirii în funcție de intrare va avea o formă asemănătoare cu cea din Figura 8.8. Se poate observa numărul mai mic de valori de pe palierele inferioare și superioare, ca urmare a distribuției normale, conform căreia valorile sunt grupate preponderent în jurul mediei. Vom urmări, prin implementarea unei cuantizări optimale, alocarea unor pași de cuantizare mai mari pentru intervalele în care semnalul de intrare ia un număr mic de valori (în cazul de față efectul ar fi lățirea palierelor inferioare și superioare), respectiv pași mai mici pentru intervalele cu mai multe valori (îngustarea palierelor centrale).

Pentru generarea pragurilor de cuantizare și a valorilor cuantizate optimale se va utiliza funcția Matlab `lloyds`. Pentru utilizarea acestei funcții în

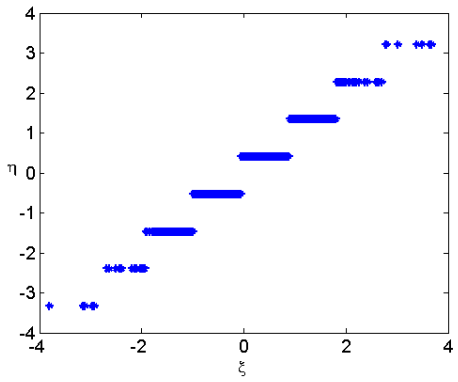


Figura 8.8: Funcția de cuantizare uniformă pentru semnalul distribuit normal.

mediul Octave este necesară încărcarea prealabilă a pachetului *communications* prin introducerea următoarei comenzi:

```
>> pkg load communications
```

Funcția `lloyds` primește ca argumente semnalul de intrare și vectorul y al valorilor cuantizate obținute anterior. Noile praguri și valori cuantizate vor fi stocate în vectorii x_n, respectiv y_n:

```
x_n(1) = min(xi);
x_n(L+1) = max(xi);

[x_n(2:L), y_n] = lloyds(xi, y);
```

Se repetă operația de cuantizare utilizând parametrii optimali, obținându-se noul semnal de ieșire:

```
for j = 1 : N
    for i = 1 : L
        if xi(j) >= x_n(i) && xi(j) <= x_n(i+1)
            eta_n(j) = y_n(i);
        end
    end
end

figure;
plot(eta_n);

figure;
plot(xi, eta_n, '*');
```

Forma semnalului de ieșire este prezentată în Figura 8.9, iar graficul funcției optimale de cuantizare este reprezentat în Figura 8.10.

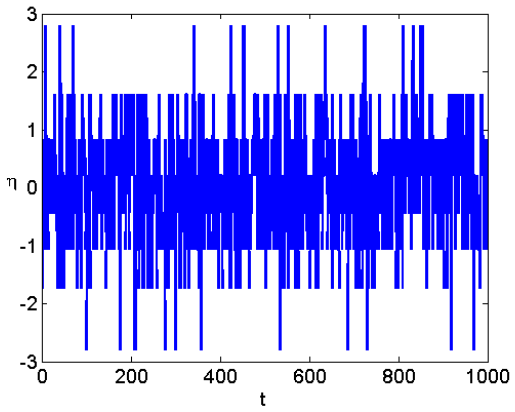


Figura 8.9: Semnalul distribuit normal după cuantizarea optimală.

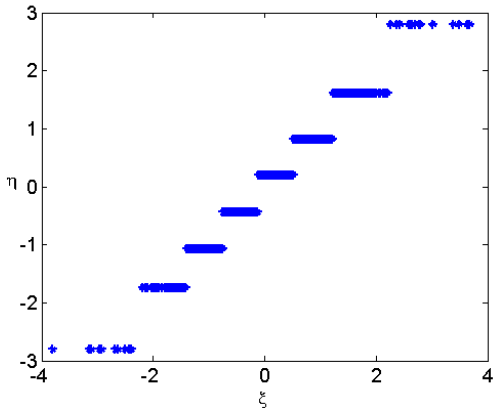


Figura 8.10: Funcția de cuantizare optimală pentru semnalul distribuit normal.

8.5 Exerciții

1. Calculați și reprezentați grafic zgomotul de cuantizare $e[n]$ în cazul cuantizării uniforme și a celei optimale. În ambele cazuri, calculați energia zgomotului.

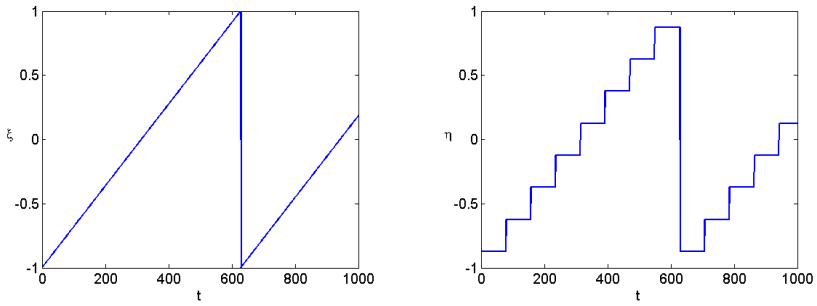
2. Calculați raportul semnal-zgomot (Signal-to-Noise Ratio - SNR) pentru semnalele cuantizate, pe baza semnalului original și a zgomotului de cuantizare, utilizând formula:

$$SNR = 10 \log \frac{\sum_{k=1}^{N} (\xi[k])^2}{\sum_{k=1}^{N} (e[k])^2} [dB] \tag{8.16}$$

3. Repetați pașii operațiilor de cuantizare pe 8 nivele pentru un semnal de tip dinte de fierăstrău.

4. Repetați pașii operațiilor de cuantizare pe 8 nivele pentru un semnal aleator cu distribuție Rayleigh. Pentru generarea a 1000 de valori ale unui astfel de semnal, având parametrul $\sigma = 1$, se poate utiliza comanda `raylrnd(1,[1,1000])`.

5. Repetați pașii operațiilor de cuantizare variind numărul de nivele de cuantizare.



(a) (b)

Figura 8.11: Semnalul de tip dinte de fierăstrău: (a) original, (b) cuantizat.

Lucrarea 9

Corelaţia semnalelor

Scopul lucrării este acela de a studia funcţiile de autocorelaţie şi intercorelaţie, precum și densitatea spectrală de putere și Teorema Wiener-Hincin în cazul unor semnale aleatoare cu densitate de probabilitate uniformă, normală şi Rayleigh.

9.1 Funcţia de autocorelaţie

Funcţia de autocorelaţie pentru un semnal aleator $\xi(t)$ se defineşte ca fiind corelaţia dintre $\xi(t_1)$ şi $\xi(t_2)$, $\forall t_1, t_2 \in \mathbb{R}$, adică dintre ξ la momentul de timp t_1 şi ξ la momentul t_2:

$$R_\xi(t_1, t_2) = \overline{\xi(t_1)\xi(t_2)} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} x_1 x_2 w_2(x_1, x_2; t_1, t_2) dx_1 dx_2 \tag{9.1}$$

În ipoteza că semnalul $\xi(t)$ este staţionar, atunci funcţia de autocorelaţie va depinde doar de diferenţa de timp dintre t_1 şi t_2:

$$R_\xi(\tau) = \overline{\xi(t)\xi(t+\tau)} \tag{9.2}$$

unde $\tau = t_2 - t_1$.

În ipoteza ergodicității semnalului, un estimat al funcţiei de autocorelaţie pentru o secvenţă aleatoare $x(n)$ este dat de formula:

$$R_\xi[m] = \frac{1}{N-m}\sum_{n=1}^{N-m} x[n]x[n+m-1], m = 1, ..., N/2 \tag{9.3}$$

9.1.1 Proprietăţile funcţiei de autocorelaţie

Funcţia de autocorelaţie are următoarele proprietăţi:

1. Funcţia de autocorelaţie este pară:

$$R_\xi(\tau) = R_\xi(-\tau)$$

2. Funcţia de autocorelaţie este maximă în origine:

$$R_\xi(0) \geq |R_\xi(\tau)|$$

3. În ipoteza că nu există componente periodice sau deterministe, valoarea funcţiei de autocorelaţie la infinit este egală cu pătratul mediei semnalului:

$$R_\xi(\infty) = \overline{\xi}^2$$

4. Media pătratică şi varianţa semnalului se obţin din funcţia de autocorelaţie astfel:

$$\overline{\xi^2} = R_\xi(0)$$
$$\sigma_\xi^2 = R_\xi(0) - R_\xi(\infty)$$

5. Dacă semnalul aleator este periodic, atunci şi funcţia lui de autocorelaţie este periodică, având aceeaşi perioadă:

$$\xi(t) = \xi(t+T) \rightarrow R_\xi(\tau) = R_\xi(\tau+T)$$

9.2 Funcţia de intercorelaţie

Fie $\xi(t)$ şi $\eta(t)$ două semnale staţionare. Funcţia de intercorelaţie între $\xi(t)$ şi $\eta(t)$ se defineşte ca fiind corelaţia dintre ξ la momentul de timp t_1 şi η la momentul t_2, astfel:

$$R_{\xi\eta}(t_1, t_2) = \overline{\xi(t_1)\eta(t_2)} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} x_1 y_2 w_2(x_1, y_2; t_1, t_2) dx_1 dy_2 \tag{9.4}$$

În ipoteza de staționaritate, funcția de intercorelație va depinde la rândul ei doar de diferenţa dintr momentele de timp t_1 şi t_2:

$$R_{\xi\eta}(\tau) = \overline{\xi(t)\eta(t+\tau)} \tag{9.5}$$

unde $\tau = t_2 - t_1$.

În aceeași ipoteză de ergodicitate, un estimat al funcţiei de intercorelaţie pentru două secvenţe aleatoare discrete $x[n]$ şi $y[n]$ este dat de formula:

$$R_{\xi\eta}[m] = \frac{1}{N-m}\sum_{n=1}^{N-m} x[n]y[n+m-1], m = 1, ..., N/2 \tag{9.6}$$

9.3 Densitatea spectrală de putere

Densitatea spectrală de putere este utilizată pentru caracterizarea din punct de vedere spectral a semnalelor aleatoare în sens larg. Considerând o variantă trunchiată a unei realizări particulare a semnalului $\xi(t)$:

$$\xi_T^{(k)}(t) = \begin{cases} \xi^{(k)}(t) & \text{dacă } |t| \leq \frac{T}{2} \\ 0 & \text{în rest} \end{cases} \tag{9.7}$$

densitatea spectrală de putere a semnalului este definită ca:

$$q_\xi(\omega) = \lim_{T\to\infty} \frac{\overline{\left|X_T^{(k)}(\omega)\right|^2}}{T} \tag{9.8}$$

unde $X_T^{(k)}(\omega)$ reprezintă Transformata Fourier a variantei trunchiate a realizării particulare $\xi^{(k)}(t)$. Trunchierea este necesară pentru ca puterea semnalului să fie finită; pentru un semnal aleator staționar aceasta poate fi infinită.

În practică, estimarea densității spectrale de putere se face pe baza Transformatei Fourier $X(\omega)$ a semnalului $\xi(t)$, estimatul reprezentând pătratul modulului transformatei raportat la numărul de eșantioane:

$$\hat{q}_\xi(\omega) = \frac{|X(\omega)|^2}{N} \tag{9.9}$$

9.4 Teorema Wiener-Hincin

Teorema Wiener-Hincin face legătura între densitatea spectrală de putere și funcția de autocorelație pentru semnalele aleatoare staționare în sens larg. Enunțul teoremei este următorul: *Densitatea spectrală de putere a unui semnal aleator staționar în sens larg este Transformata Fourier a funcției sale de autocorelație*:

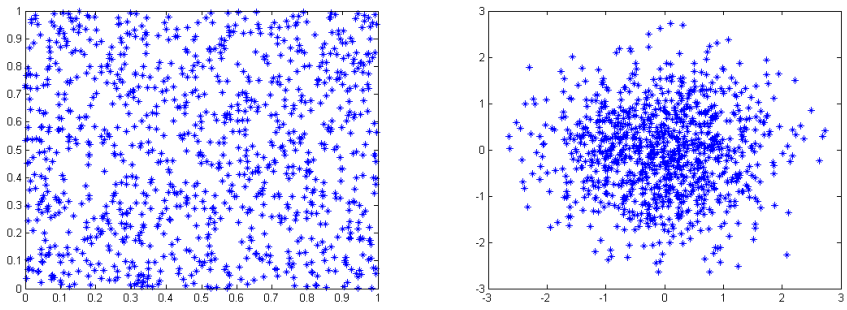
$$q_\xi(\omega) = \int_{-\infty}^{+\infty} R_\xi(\tau) e^{-j\omega\tau} d\tau \tag{9.10}$$

$$R_\xi(\tau) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} q_\xi(\omega) e^{j\omega\tau} d\omega \tag{9.11}$$

Astfel, conform teoremei Wiener-Hincin, funcțiile $R_\xi(\tau)$ și $q_\xi(\omega)$ reprezintă *perechi Fourier*.

9.5 Diagrama în spaţiul stărilor

Dacă ne interesează să apreciem calitativ dependența dintre două variabile aleatoare, ξ şi η, atunci când nu cunoaștem funcțiile de densitate de probabilitate care caracterizează perechea de variabile, se poate folosi *diagrama în spaţiul stărilor*. Aceasta reprezintă un nor de puncte într-un spaţiu bidimensional, format din perechi $(\xi_{(i)}, \eta_{(i)})$ ale realizărilor particulare ale perechii de variabile. Forma acestui nor furnizează informații despre corelaţia dintre cele două semnale: un nor format din puncte haotic distribuite în spațiu indică variabile independente din punct de vedere statistic (vezi Figura 9.1), pe când unul *ordonat* indică o anumită dependență între cele două variabile.



(a) (b)

Figura 9.1: Diagrama în spațiul stărilor pentru două variabile independente distribuite (a) uniform; (b) normal.

Diagrama în spaţiul stărilor asociată unui semnal aleator $\xi(t)$ este o reprezentare bidimensională a unei variabile aleatoare care reprezintă eşantionul de la momentul t al semnalului în funcţie de variabila aleatoare asociată semnalului la momentul $t-p$, norul fiind astfel alcătuit din puncte aflate la coordonatele $(\xi(t), \xi(t-p))$. Reprezentarea se face pe o durată de observaţie T aleasă sau dată.

Cu ajutorul diagramei în spaţiul stărilor putem vizualiza legătura statistică între cele două variabile aleatoare pentru diferite valori de "întârziere" p, obţinând o evaluare calitativă referitoare la corelaţia conţinută în semnal şi la natura (aleatoare sau deterministă) a semnalului.

Pentru semnalele deterministe aspectul diagramei corespunde legăturilor funcţionale dintre eşantioanele semnalului, diagrama fiind o reprezentare grafică a acestora. Pentru semnalele aleatoare aspectul diagramei se prezintă sub forma unei mulţimi de puncte care pot fi incluse într-un contur închis cu o anumită formă. Această formă ne poate da informaţii cu privire la corelaţia dintre eşantioane.

9.6 Desfăşurarea lucrării

Se vor genera trei secvenţe de $N = 1000$ valori aleatoare, având distribuție uniformă, normală, respectiv Reyleigh, cu media şi dispersia cunoscute (vezi Lucrarea 7). Pentru fiecare din cele trei secvenţe aleatoare se va reprezenta grafic diagrama în spaţiul stărilor pentru diverse valori ale lui p. Acest lucru se va realiza folosind următoarea instrucţiune Matlab: `plot( x(p+1:N), x(1:N-p), '*' )`.

Pentru fiecare din cele trei secvenţe aleatoare se va determina şi reprezenta grafic funcţia de autocorelaţie, folosind funcţia Matlab `xcorr`.

Pentru a utiliza funcția în mediul Octave este necesară încărcarea prealabilă a pachetului *signal* prin introducerea în terminal a următoarei comenzi:

```
>>pkg load signal
```

Afișarea graficului funcției de autocorelație pentru o secvență `x` se face utilizând comanda `plot(xcorr(x, N/2, 'unbiased'))`. Un exemplu de grafic pentru un semnal aleator cu distribuție normală (Gaussiană) este prezentat în Figura 9.2. Valoarea maximă din origine se regăsește la mijlocul graficului.

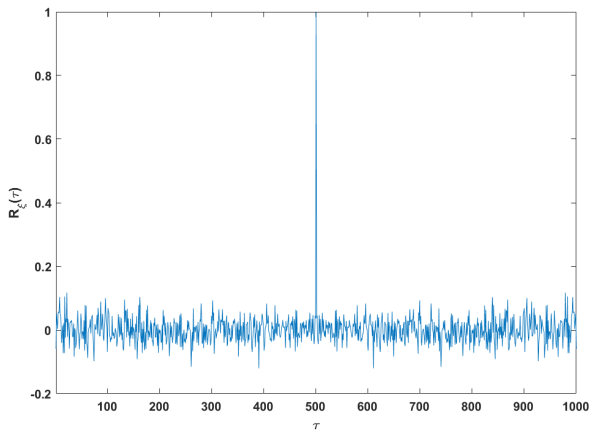


Figura 9.2: Funcţia de autocorelaţie pentru un semnal aleator distribuit normal.

Să se implementeze o funcţie de calcul al estimatului funcției de autocorelație, pe baza formulei:

$$R_\xi[m] = \frac{1}{N-m} \sum_{n=1}^{N-m} x[n]x[n+m-1], m = 1, ..., N/2$$

Să se determine și reprezinte grafic estimatul funcției de intercorelație pentru două dintre secevenţele aleatoare generate anterior, folosind funcția

Matlab `xcorr`. Să se implementeze o funcție de calcul a acestui estimat pe baza formulei:

$$R_{\xi\eta}[m] = \frac{1}{N-m} \sum_{n=1}^{N-m} x[n]y[n+m-1], m = 1, ..., N/2$$

9.7 Exerciții

1. Să se genereze 1000 de eșantioane ale unui semnal sinusoidal $x(t)$, de amplitudine $A = 4$, frecvență $\omega = 0.05$ și fază $\phi = 0$. Să se suprapună zgomot peste acest semnal. Reprezentați diagrama în spațiul stărilor pentru acest semnal, pentru diferite valori ale parametrului p. Calculați și reprezentați grafic funcția de autocorelație, înainte și după suprapunerea zgomotului.

2. Atât pentru semnalele generate pe parcursul lucrării, cât și pentru un semnal dreptunghiular, calculați și reprezentați grafic transformata Fourier a funcției de autocorelație, respectiv estimatul densității spectrale de putere.

3. Pentru perechi de semnale aleatoare generate pe parcursul lucrării, calculați și reprezentați grafic funcția de intercorelație.

Lucrarea 10

Filtrul adaptat la semnal

Filtrul adaptat la semnal, studiat în cadrul acestei lucrări, este un tip aparte de filtru, proiectarea acestuia făcându-se prin specificarea funcţiei pondere în domeniul timp, nu prin specificarea răspunsului în frecvență, cum este cazul filtrului trece-jos. Acest filtru se foloseşte de regulă pentru detecţia unor semnale sau a anumitor forme în semnal, de exemplu într-un lanţ de transmisiune, mai exact, atunci când se cunoaşte forma semnalului original transmis, care la recepţie este afectat de zgomot.

10.1 Proiectarea filtrului adaptat la semnal

Pentru un semnal determinist de durată finită, $s(t)$, se defineşte filtrul adaptat la semnal ca fiind filtrul liniar invariant în timp care are următoarea funcţie pondere:

$$h(t) = K \cdot s(-(t - t_0)) \tag{10.1}$$

unde K şi t_0 sunt constante reale oarecare, cu constrângerea că t_0 trebuie astfel ales încât filtrul să fie *cauzal*.

Un exemplu de filtru adaptat la semnal este prezentat în Figura 10.1.

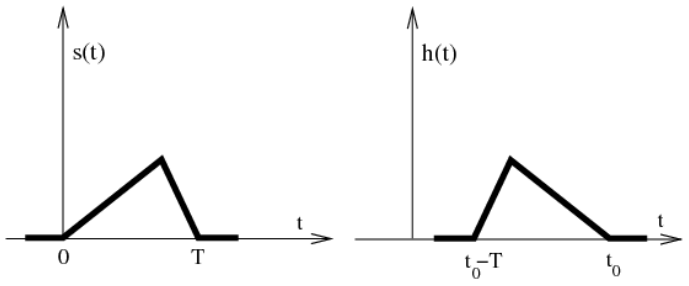


Figura 10.1: Exemplu de filtru adaptat la semnal - semnalul original (stânga) şi funcția pondere a filtrului (dreapta).

Răspunsul în frecvență al filtrului adaptat la semnal este transformata Fourier a funcției pondere:

$$H(\omega) = \mathcal{F}\{h(t)\}(\omega) = K \int_{-\infty}^{+\infty} h(t) \cdot e^{-j\omega t} dt = K \int_{-\infty}^{+\infty} s(-(t-t_0)) \cdot e^{-j\omega t} dt \tag{10.2}$$

Făcând schimbarea de variabilă $\tau = -(t - t_0)$ obţinem:

$$H(\omega) = -K \int_{+\infty}^{-\infty} s(\tau) \cdot e^{-j\omega(t_0-\tau)} d\tau = K \cdot e^{-j\omega t_0} \int_{-\infty}^{+\infty} s(t) \cdot e^{j\omega\tau} d\tau \tag{10.3}$$

$$H(\omega) = KS^*(\omega)e^{-j\omega t_0} \tag{10.4}$$

unde $S(\omega)$ este transformata Fourier a semnalului $s(t)$, iar $S^*(\omega)$ complex conjugata acesteia.

O proprietate foarte importantă a acestui filtru este faptul că maximizează raportul semnal/zgomot.

10.2 Desfăşurarea lucrării

Generaţi un semnal de 1000 de eşantioane, care conţine un impuls dreptunghiular de lăţime 100 de eşantioane, ca cel din Figura 10.2, utilizând următoarea secvență Matlab:

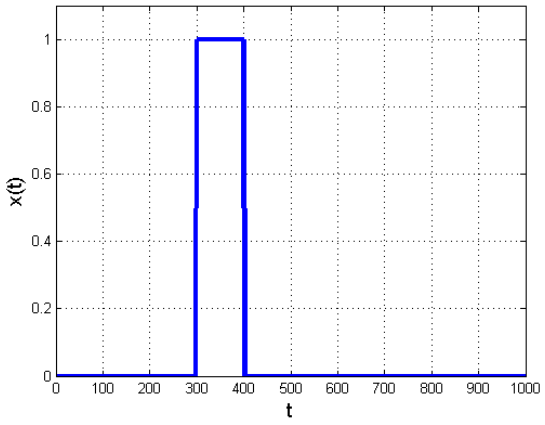


Figura 10.2: Semnal de tip impuls dreptunghiular.

```
x = zeros(1,1000);
x(300:400) = 1;
```

Peste acest semnal se va suprapune zgomot, utilizând funcţia Matlab `rand`:

```
x = x + rand(1,1000);
```

Semnalul afectat de zgomot va fi de forma celui din Figura 10.3.

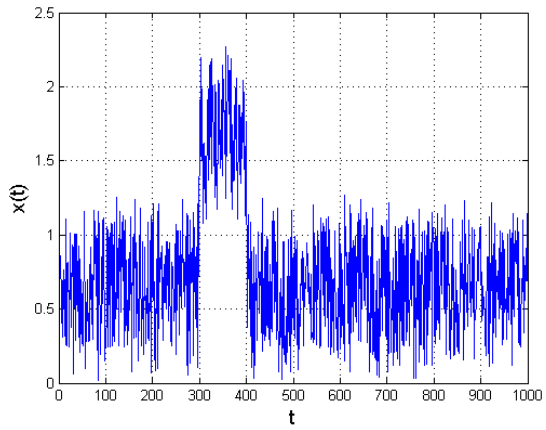


Figura 10.3: Semnalul afectat de zgomot.

Ştiind că semnalul $x(t)$ recepţionat conţine un impuls dreptunghiular de lățime 100 eşantioane, vom proiecta un filtru adaptat la semnal, având funcţia pondere $h(t)$ ca cea din Figura 10.4, conținând un impuls de 100 de eșantioane (lățimea impulsului ce se dorește a fi detectat). Pentru aceasta se va utiliza secvenţa Matlab următoare:

```
h = zeros(1,300);
h(100:200)=1;
```

Semnalul $y(t)$ de la ieşirea filtrului adaptat la semnal (vezi Figura 10.5) se va calcula cu ajutorul funcţiei Matlab `filter`, care va realiza convoluţia dintre semnalul $x(t)$ şi funcţia pondere $h(t)$ a filtrului.

```
y = filter(h, 1, x);
```

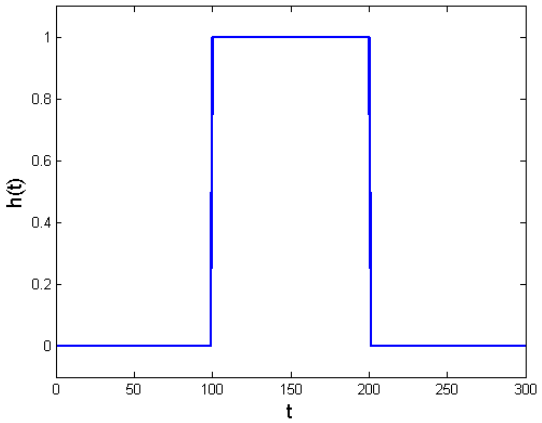


Figura 10.4: Funcţia pondere a filtrului adaptat la semnal.

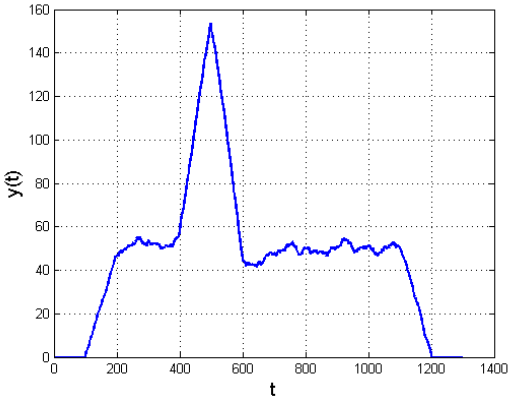


Figura 10.5: Semnalul $y(t)$ de la ieşirea filtrului adaptat la semnal.

Valoarea maximă din semnalul $y(t)$ indică poziţia în care filtrul adaptat la semnal a detectat un impuls dreptunghiular similar cu funcţia pondere, adică locul în care potrivirea dintre semnalul $x(t)$ de la intrarea filtrului şi "semnalul" $h(t)$ a fost cea mai bună. Ieșirea filtrului adaptat la semnal reprezintă funcția de intercorelație dintre semnalul de la intrare și semnalul căutat. Semnalul original ar putea fi recuperat prin identificarea punctelor de maxim ale semnalului $y(t)$.

10.3 Exerciții

1. Generaţi un semnal care conţine câteva impulsuri de formă dreptunghiulară, de aceeaşi lăţime. Suprapuneţi zgomot peste acest semnal, folosind funcțiile Matlab `rand` şi `randn`. Filtraţi semnalul cu un filtru adaptat la semnal, proiectat în prealabil. Vizualizaţi rezultatul. Ce observaţi?

2. Aceeaşi problemă ca mai sus, pentru cazul în care semnalul conţine impulsuri de formă triunghiulară. Pentru generarea unui impuls triunghiular $\Lambda(t)$ simetric de lățime L, centrat în momentul c, puteți utiliza formula:

$$\Lambda(t) = \begin{cases} 1 - \frac{2|t-c|}{L} & \text{dacă } t \in [c - \frac{L}{2}, c + \frac{L}{2}] \\ 0 & \text{în rest} \end{cases} \tag{10.5}$$

Un exemplu de impuls triunghiular de lățime $L = 100$ eșantioane, centrat în momentul $c = 150$, este prezentat în Figura 10.6.

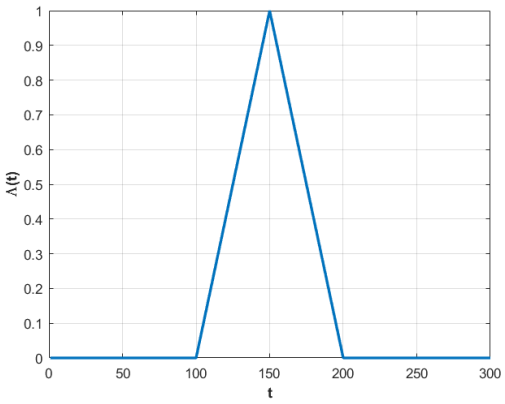


Figura 10.6: Semnal de tip impuls triunghiular.

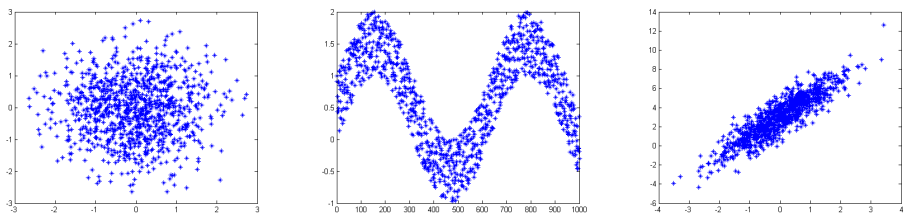
Lucrarea 11

Dreapta de regresie și analiza componentelor principale

În această lucrare ne propunem să studiem cantitativ dependenţa statistică dintre două variabile aleatoare, concret să determinăm gradul de dependenţă liniară dintre ele prin calcularea coeficientului de corelaţie şi determinarea dreptei de regresie. De asemenea, ne propunem studiul analizei componentelor principale, care permite reducerea dimensiunilor unui semnal multidimensional.

11.1 Dreapta de regresie

Reamintim faptul că diagrama în spaţiul stărilor pentru două variabile aleatoare permitea aprecierea calitativă a dependenţei dintre acestea. În Figura 11.1 puteţi observa forma norului de puncte pentru trei cazuri: a) atunci când cele două variabile aleatoare sunt independente din punct de vedere statistic; b) când între variabile există o anumită dependență funcțională, dar acestea sunt decorelate şi c) când variabilele sunt corelate, şi deci sunt şi dependente statistic.



(a) (b) (c)

Figura 11.1: Forma norului de puncte pentru două variabile (a) independente statistic; (b) dependente dar decorelate şi (c) dependente şi corelate.

Dacă diagrama în spaţiul stărilor indică faptul că între cele două variabile aleatoare ar exista o dependență aproximativ liniară, de forma celei din Figura 11.1c) atunci se poate determina dreapta de regresie (vezi Figura 11.3) care ar putea aproxima cel mai bine această dependență liniară.

Ecuația dreptei de regresie, care minimizează eroarea pătratică medie de aproximare a dependenței liniare dintre ξ și η este:

$$\hat{\eta} = \frac{K_{\xi\eta}}{\sigma_\xi^2}(\xi - \overline{\xi}) + \overline{\eta} \tag{11.1}$$

unde $K_{\xi\eta}$ reprezintă covariaţia dintre cele două variabile, $\overline{\xi}$ este media statistică a lui ξ, iar $\overline{\eta}$ media lui η.

Covariaţia dintre ξ şi η reprezintă momentul centrat mixt de ordinul doi, $K_{\xi\eta} = \overline{(\xi - \overline{\xi})(\eta - \overline{\eta})}$ şi diferă de corelaţie doar printr-o constantă: $K_{\xi\eta} = R_{\xi\eta} - \overline{\xi}\overline{\eta}$. Două variabile se numesc *decorelate* atunci când $K_{\xi\eta} = 0$. Două variabile aleatoare independente sunt implicit decorelate.

11.1.1 Coeficientul de corelaţie

Pornind de la expresia erorii pătratice medii minime dintre variabila aleatoare η şi cea care o aproximează cel mai bine, $\hat{\eta}$:

$$\varepsilon_{min} = \overline{(\eta - \hat{\eta})^2} = \sigma_\eta^2 \left[1 - \left(\frac{K_{\xi\eta}}{\sigma_\xi \sigma_\eta}\right)^2\right] \tag{11.2}$$

se definește coeficientul de corelație, notat cu $\rho_{\xi\eta}$:

$$\rho_{\xi\eta} = \frac{K_{\xi\eta}}{\sigma_\xi \sigma_\eta} \tag{11.3}$$

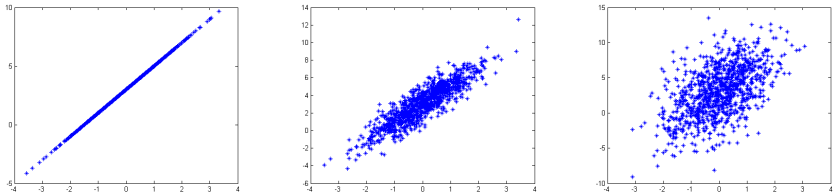
şi care reprezintă valoarea normată a covariației dintre cele două variabile aleatoare. În acest fel, coeficientul de corelaţie permite cuantificarea absolută a gradului de dependenţă liniară dintre ξ şi η.

Dacă rescriem expresia erorii pătratice medii ca fiind:

$$\varepsilon_{min} = \sigma_\eta^2(1 - \rho_{\xi\eta}^2) \tag{11.4}$$

se poate observa că $\rho_{\xi\eta}$ trebuie să fie de modul subunitar, pentru ca eroarea pătratică să fie pozitivă (fiind o sumă de cantităţi pozitive): $|\rho_{\xi\eta}| \leq 1$. Cu cât $|\rho_{\xi\eta}|$ este mai aproape de 1, cu atât eroarea de aproximare a lui η cu $\hat{\eta}$ e mai mică, deci gradul de dependenţă liniară dintre variabile este mai mare. Dacă $|\rho_{\xi\eta}| = 1$ atunci $\varepsilon = 0$ – norul de puncte este chiar o dreaptă, dependenţa dintre cele două variabile fiind perfect liniară.

În Figura 11.2 puteţi observa forma norului de puncte, sau a diagramei în spaţiul stărilor, pentru două variabile aleatoare ξ şi η, pentru diverse valori ale gradului de dependenţă liniară dintre acestea.



(a) $\rho_{\xi\eta} = 1$ (b) $\rho_{\xi\eta} = 0.9$ (c) $\rho_{\xi\eta} = 0.5$

(d) $\rho_{\xi\eta} = -0.5$ (e) $\rho_{\xi\eta} = -0.9$ (f) $\rho_{\xi\eta} = -1$

Figura 11.2: Forma diagramei în spaţiul stărilor pentru ξ şi η, pentru diverse valori ale coeficientului de corelaţie.

11.2 Analiza componentelor principale

Analiza componentelor principale (*Principal Component Analysis* - PCA) reprezintă un algoritm utilizat în general pentru reducerea dimensionalității seturilor de date vectoriale (multidimensionale). Algoritmul generează un set de axe ortogonale utilizat pentru reprezentarea optimă a datelor, noile axe fiind alese astfel încât să se afle pe direcțiile ortogonale de maximă varianță a datelor.

Un semnal discret multidimensional X poate fi reprezentat prin intermediul unei matrici de dimensiune $n \times m$:

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{pmatrix} \tag{11.5}$$

unde n reprezintă numărul de eșantioane m-dimensionale ale semnalului. Exemple de semnale multidimensionale includ semnalele audio stereo ($m = 2$) sau mulțimi de pixeli dintr-o imagine color ($m = 3$).

Sistemul ortogonal de axe generat de algoritmul PCA este determinat de vectorii proprii ai matricii de covariaţie a datelor vectoriale centrate în zero. Etapele algoritmului sunt următoarele:

1. Obţinerea datelor vectoriale centrate în zero prin scăderea valorii medii

a fiecărei componente din componenta respectivă:

$$X_c = X - 1\overline{X} \tag{11.6}$$

unde 1 reprezintă un vector coloană de lungime n, iar $\overline{X}$ este un vector linie dat de

$$\overline{X} = (\overline{x}_1, \overline{x}_2, \cdots, \overline{x}_m) \tag{11.7}$$

în timp ce $\overline{x}_j$ este media vectorului coloană x_j, determinată prin

$$\overline{x}_j = \frac{1}{n}\sum_{k=1}^{n} x_j(k). \tag{11.8}$$

Notăm elementele matricii X_c cu $x_{c_{ij}}$, în mod similar elementelor matricii X;

2. Calculul matricii de covariaţie a datelor centrate:

$$K_X = \begin{pmatrix} K_{11} & K_{12} & \cdots & K_{1m} \\ K_{21} & K_{22} & \cdots & K_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ K_{m1} & K_{m2} & \cdots & K_{mm} \end{pmatrix} \tag{11.9}$$

unde K_{jk} este covariaţia dintre componentele j şi k ale datelor centrate:

$$K_{jk} = \overline{\big((x_{c_{ij}} - \overline{x}_{c_j})(x_{c_{ik}} - \overline{x}_{c_k})\big)} = \overline{x_{c_{ij}} x_{c_{ik}}}, \tag{11.10}$$

3. Calculul vectorilor proprii v_j $(j = 1, 2, \cdots, m)$ ai matricii de covariaţie K_X, aceștia reprezentând noul sistem ortogonal de axe. Direcţia vectorului propriu asociat celei mai mari valori proprii reprezintă direcţia pe care varianţa datelor iniţiale este maximă, direcţia vectorului propriu (perpendicular pe primul vector propriu) corespunzător celei de-a doua valori proprii, ca mărime, reprezintă direcţia pe care datele au a doua cea mai mare varianță, etc.

11.3 Desfăşurarea lucrării

Să se genereze un semnal aleator $x(n)$ cu distribuţie normală, şi să se construiască un semnal $y(n)$ de forma: $y(n) = a \cdot x(n) + b + k \cdot \xi$, unde ξ este o variabilă aleatoare distribuită normal, iar k un factor de ponderare. Să se calculeze coeficientul de corelaţie dintre cele două semnale, folosind funcţia Matlab `corrcoef` şi să se determine dreapta de regresie, folosind funcţia `robustfit`. Să se reprezinte grafic diagrama în spaţiul stărilor, pentru cele două semnale. Să se reprezinte grafic dreapta de regresie, în aceeaşi figură cu norul de puncte (vezi Figura 11.3). Rezolvările cerințelor se regăsesc în codul de mai jos:

```
x = randn(1,10000);
y = 2*x + 3 + randn(1,10000);
corrcoef(x,y)
b = robustfit(x,y,'ols')
figure;
plot(x, y, '*');
hold on;
plot(x, b(2)*x+b(1),'r');
```

Considerând $x(n)$ și $y(n)$ ca fiind cele două componente ale unui semnal multivariat $z(n)$, să se calculeze și afișeze cele două componente principale, utilizând funcția `pca`.

Pentru realizarea analizei componentelor principale în mediul Octave este necesară încărcarea pachetului *tisean* prin introducerea următoarei comenzi:

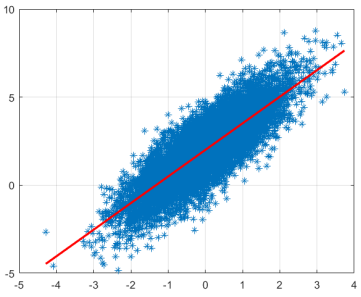
```
>>pkg load tisean
```

Primul pas înainte de aplicarea funcției îl reprezintă transpunerea vectorilor `x` și `y` și concatenarea lor în matricea `z`.

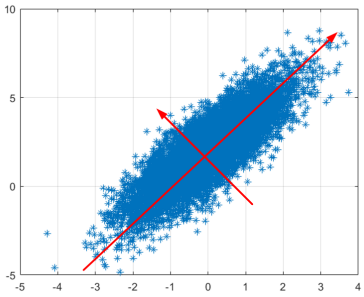
```
z = [x' y'];
[eigval,eigvect] = pca(z);
```

Funcția returnează vectorii proprii (`eigvect`) și valorile proprii (`eigval`), acestea din urmă reprezentând variațele asociate fiecărei componente principale. In vederea afișării, vectorii proprii sunt scalați astfel încât lungimea lor să fie proporțională cu varianța corespunzătoare, aceștia fiind plasați apoi in centrul norului de puncte, obținut ca media valorilor semnalului bidimensional `z`. Cele două componente principale, `pc1` și `pc2` sunt afișate apoi în aceeași figură cu norul de puncte (vezi Figura 11.3(b)).

```
sigma1_vect = eigvect(1,:)*eigval(1,2)*2;
sigma2_vect = eigvect(2,:)*eigval(2,2)*2;

m = mean(z);

pc1(1,:) = m + sigma1_vect;
pc1(2,:) = m - sigma1_vect;

pc2(1,:) = m + sigma2_vect;
pc2(2,:) = m - sigma2_vect;

figure;
plot(x, y, '*');
hold on;
plot(pc1(:,1), pc1(:,2), 'r');
plot(pc2(:,1), pc2(:,2), 'r');
```



(a) Dreapta de regresie.



(b) Componentele principale.

Figura 11.3: Dreapta de regresie, respectiv componentele principale, suprapuse peste diagrama în spațiul stărilor.

11.3.1 Componentele principale ale unei imagini color

În cadrul acestei secțiuni, algoritmul PCA va fi aplicat asupra unei imagini color, obținându-se cele trei componente principale, ordonate descrescător în funcție de varianță. Imaginea pe care se va aplica algoritmul este Lena, o imagine color standard de test. Imaginea este prezentată în Figura 11.4, iar componentele R, G, B individuale, reprezentate ca imagini în nivele de gri, sunt prezentate în Figura 11.5.

Imaginile sunt semnale reprezentate matematic ca funcții de două variabile, acestea fiind coordonatele în plan ale pixelilor (punctelor) imaginii:

- Imaginile în nivele de gri pot fi modelate ca o funcție $f(x, y) : \mathbb{R}^2 \to \mathbb{R}$, valorile funcției reprezentând luminanțele pixelilor din imagine.
- Imaginile color pot fi reprezentate ca $f(x, y) : \mathbb{R}^2 \to \mathbb{R}^3$, valorile funcției reprezentând vectori cu 3 componente dintr-un spațiu de culoare. Spațiul cel mai des utilizat pentru achiziția și reprezentarea imaginilor color este RGB (Red Green Blue), culorile fiind compuse

Figura 11.4: Imaginea color Lena.

(a) Componenta R. (b) Componenta G. (c) Componenta B.

Figura 11.5: Componentele de roșu, verde și albastru ale imaginii color Lena.

dintr-o componentă de roșu, una de verde și una de albastru. Alte spații de culoare, cum ar fi Lab sau YUV, conțin o componentă de luminanță (reprezentând intensitatea luminoasă a pixelilor) și două componente de crominanță (culoarea propriu-zisă).

Secvența de citire a imaginii din fișier și de afișare a imaginii color, precum și a componentelor de roșu, verde și albastru, este prezentată mai jos. Valorile pixelilor, reprezentate ca întregi pe 8 biți (0...255), sunt convertite la valori reale în intervalul $[0, 1]$.

```
img = double(imread('lena.png'));
img = img/255;
figure;
imshow(img);

figure;
subplot(1, 3, 1), imshow(img(:,:,1));
subplot(1, 3, 2), imshow(img(:,:,2));
subplot(1, 3, 3), imshow(img(:,:,3));
```

Înainte de aplicarea PCA, componentele imaginii sunt transformate în vectori coloană, rezultând o matrice cu trei coloane, fiecare dintre ele conținând pixelii aferenți uneia din componentele de roșu, verde, respectiv albastru:

```
[m, n, o] = size(img);
img_vectors = reshape(img, m*n, 3);
[eigval, eigvect] = pca(img_vectors);
```

Valorile pixelilor din noul spațiu de reprezentare sunt scalate în intervalul $[0, 1]$, apoi sunt rearanjate sub forma unei imagini cu 3 componente, utilizând secvența de mai jos:

```
img_pca_vectors = (eigvect - min(min(eigvect)))/...
    (max(max(eigvect)) - min(min(eigvect)));
img_pca = reshape(img_pca_vectors, m, n, o);

figure;
subplot(1, 3, 1), imshow(img_pca(:,:,1));
subplot(1, 3, 2), imshow(img_pca(:,:,2));
subplot(1, 3, 3), imshow(img_pca(:,:,3));
```

Cele trei componente principale rezultate sunt prezentate în Figura 11.6. Se poate observa că prima componentă corespunde în principiu unei imagini de luminanță, reprezentând informația legată de intensitatea luminoasă, în timp ce celelalte două componente ar putea fi asimilate unor imagini de crominanță (conținând informația de culoare). Prin urmare, prima componentă are cea mai mare variantă, iar rezultatul aplicării algoritmului PCA pe imaginea color corespunde proprietății sistemului vizual uman de a fi mai sensibil la variațiile de luminanță decât la cele de crominanță. În plus, componentele rezultate sunt decorelate, în timp ce în spațiul original de reprezentare (RGB), acestea erau puternic corelate.

(a) Componenta 1. (b) Componenta 2. (c) Componenta 3.

Figura 11.6: Componentele principale ale imaginii color Lena.

11.4 Exerciții

1. Repetaţi calculul coeficientului de corelație, dreptei de regresie și componentelor principale pentru diverse distribuţii ale semnalelor $x(n)$ şi $y(n)$, variind factorul k de ponderare și atribuind diverse valori constantelor a şi b.

2. Calculaţi dreapta de regresie folosind celelalte metode disponibile pentru funcţia `robustfit`: Andrews, Cauchy, Talwar etc.

3. Implementaţi o funcţie care să calculeze coeficientul de corelaţie, pe baza formulei sale de definiţie, folosind funcțiile Matlab `mean` și `var`.

Lucrarea 12

Filtrarea inversă

Filtrarea inversă (sau de restaurare) se referă la o clasă de tehnici prin care se încearcă restaurarea unui semnal degradat, pe baza modelării degradării acestuia. Degradarea de care suferă semnalul nu se referă la prezența zgomotului (caz în care se aplică filtrele prezentate în lucrările anterioare), ci la "deformarea" semnalului la momentul achiziției. De exemplu, un semnal dreptunghiular va apărea ca fiind unul aproape triunghiular pe ecranul unui osciloscop incapabil de a reda tranzițiile bruște ale semnalului, caracterizate de frecvențe foarte înalte.

În această lucrare de laborator vom discuta trei abordări: filtrul invers, filtrul invers cu constrângeri și filtrul Wiener.

12.1 Modelul de degradare

Vom presupune că semnalul original, pe care nu îl cunoaștem, dar pe care dorim să îl refacem, a fost degradat conform unui model de degradare cunoscut. În cazul unui model simplu de degradare, putem considera cazul din Figura 12.1, în care semnalul original a suferit o convoluție cu un sistem cunoscut sau modelabil $H(\omega)$, iar peste rezultatul convoluției s-a suprapus zgomot, într-un mod aditiv. Concret, semnalul original $f(t)$ a suferit anumite modificări (cum ar fi atenuarea, în domeniul frecvență similară cu o filtrare trece-jos), modelabile matematic printr-un model de degradare liniară (12.1). Peste semnalul $f'(t)$ s-a suprapus zgomot, $r(t)$, obținându-se semnalul degradat $f"(t)$. Pornind de la aceasta, ne propunem să determinăm un filtru de restaurare care să ne permită refacerea semnalului $f(t)$ original într-o cât mai bună măsură.

12.1.1 Modelul de degradare liniară

Modelul de degradare liniară, cel mai simplu model de degradare, este reprezentat matematic printr-o convoluție:

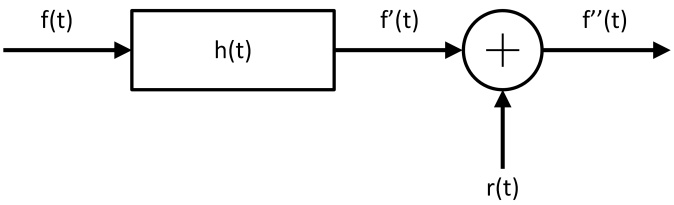


Figura 12.1: Model de degradare.

$$f'(t) = h(t) \star f(t) = \int_{-\infty}^{+\infty} h(\tau) f(t-\tau) d\tau = \int_{-\infty}^{+\infty} h(t-\tau) f(\tau) d\tau \qquad (12.1)$$

unde funcția $h(t)$ poate reprezenta, de exemplu, funcția pondere a unui sistem real, ne-ideal, de achiziție de semnal.

Vom considera impulsul dreptunghiular $f(t)$ din Figura 12.2, reprezentat de o secvență discretă.

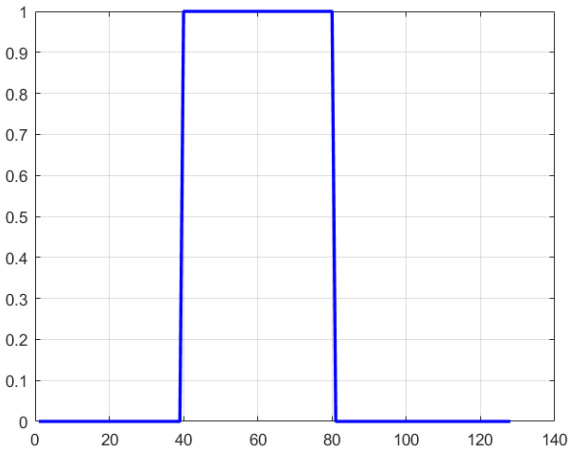
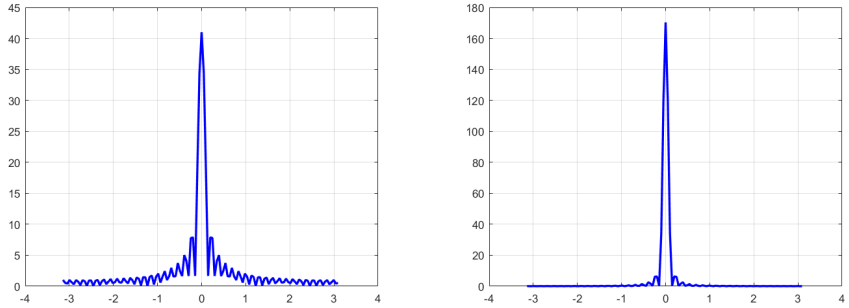


Figura 12.2: Impuls dreptunghiular.

```
N = 128;
f = zeros(1, N);
f(40:80) = 1;
figure;
plot(f);
```

Vom calcula spectrul său Fourier de amplitudini $|F(\omega)|$ și densitatea spectrală de putere (DSP), pe care o vom estima ca pătratul spectrului



(a) $|F(\omega)|$ (b) $|F(\omega)|^2$

Figura 12.3: Spectrul Fourier de amplitudini și estimatul densității spectrale de putere ale impulsului dreptunghiular.

de amplitudini ($|F(\omega)|^2$). Pentru impulsul dreptunghiular generat, aceste funcții sunt prezentate în Figura 12.3.

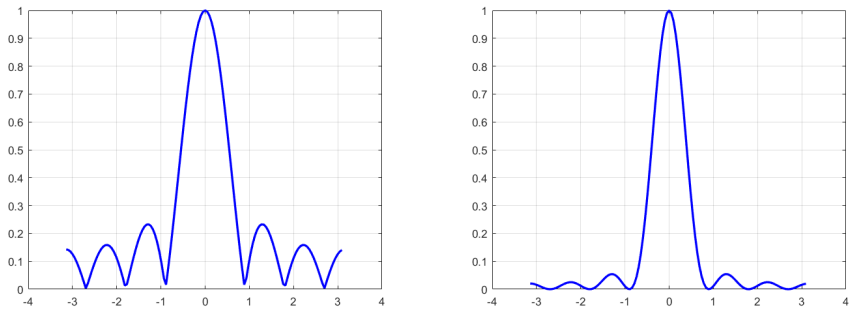
```
F = fft(f);
F_dsp = abs(F).^2;
w = -pi : (2*pi)/N : pi - (2*pi)/N;
figure;
plot(w, abs(fftshift(F)));
figure;
plot(w, fftshift(F_dsp));
```

Degradarea o vom considera ca fiind o filtrare trece-jos. Prin urmare, acest semnal va fi degradat, mai întâi, cu un filtru de mediere (de exemplu cu o fereastră de lungime 7) caracterizat spectral de funcțiile $|H(\omega)|$ şi $|H(\omega)|^2$, prezentate în Figura 12.4. Funcția pondere a unui astfel de filtru este următoarea:

$$h(t) = \left[\frac{1}{7}\ \frac{1}{7}\ \frac{1}{7}\ \frac{1}{7}\ \frac{1}{7}\ \frac{1}{7}\ \frac{1}{7}\right] \tag{12.2}$$

```
h = 1/7 * ones(1, 7);
H = fft(h, N);
H_dsp = abs(H).^2;
figure;
plot(w, abs(fftshift(H)));
figure;
plot(w, fftshift(H_dsp));
```

Rezultatul degradării, pentru cazul considerat, al convoluţiei $h(t) \star f(t)$ în domeniul timp sau al produsului $H(\omega)F(\omega)$ în domeniul frecvenţă, este



(a) $|H(\omega)|$ (b) $|H(\omega)|^2$

Figura 12.4: Răspunsul în frecvență al filtrului de mediere (a), respectiv densitatea spectrală de putere a funcției pondere asociate (b).

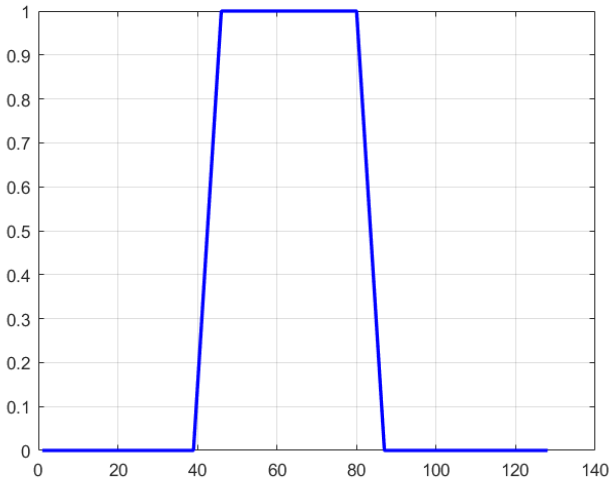


Figura 12.5: Rezultatul filtrării de mediere.

ilustrat în Figura 12.5. Se poate observa faptul că variaţiile sunt mai "line", semnalul transformându-se dintr-unul dreptunghiular într-unul trapezoidal.

```
f_degr = filter(h, 1, f);
figure;
plot(f_degr);
```

În domeniul spectral, putem scrie:

$$F''(\omega) = H(\omega)F(\omega) + R(\omega) \quad \forall\omega$$

Zgomotul este de regulă considerat ca fiind zgomot alb, independent de

semnalul util f. Despre zgomot cunoaștem (de fapt, estimăm) energia sa totală E_r:

$$E_r = \int_{-\infty}^{+\infty} r^2(t)dt \tag{12.3}$$

Problema de rezolvat: pe baza ipotezelor privitoare la $h(t)$ și E_r ne propunem să proiectăm un filtru de restaurare liniar și invariant în timp, $q(t)$, având răspunsul în frecvență $Q(\omega)$, care să aibă ca ieșire un semnal $\widehat{f}(t)$ cât mai apropiat de semnalul original $f(t)$.

12.2 Filtrul invers de restaurare

Dacă nu se ia în calcul componenta de zgomot suprapusă după degradare, răspunsul în frecvență al filtrului invers de restaurare reprezintă chiar funcția inversă a funcției răspuns în frecvență a filtrului de degradare:

$$Q(\omega) = H^{-1}(\omega), \forall\omega \tag{12.4}$$

Rezultă că spectrul Fourier al semnalului restaurat va fi dat de:

$$\widehat{F}(\omega) = Q(\omega)F''(\omega) = F(\omega) + H^{-1}(\omega)R(\omega), \forall\omega \tag{12.5}$$

Semnalul restaurat este cel original peste care se suprapune o componentă parazită, cauzată de zgomot. H are zerouri în banda utilă a semnalului (frecvențe ω_i pentru care $H(\omega_i) \approx 0$), prin urmare H^{-1} va avea poli, ceea ce rezultă într-o amplificare puternică a componentelor spectrale ale zgomotului egale sau în preajma acestor frecvențe ω_i.

12.2.1 Restaurarea prin filtrare inversă, în lipsa zgomotului

Aplicând relația (12.4), obținem o restaurare directă a semnalului $f(t)$, după recuperarea acestuia prin transformata Fourier inversă. În Figura 12.6 se poate observa cum, prin inversare, zerourile răspunsului în frecvență $H(\omega)$ au devenit poli ai funcției $Q(\omega)$ - "vârfurile" ce reprezintă amplificări foarte mari pentru anumite frecvențe.

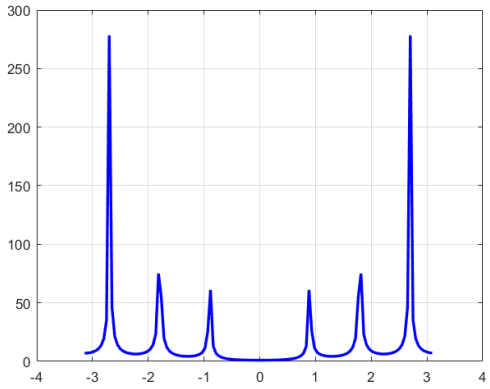


Figura 12.6: Funcția inversă de restaurare $|Q(\omega)|$.

Rezultatul filtrării inverse este prezentat în Figura 12.7. Se poate observa refacerea perfectă a semnalului original, așa cum era acesta înainte de degradare.

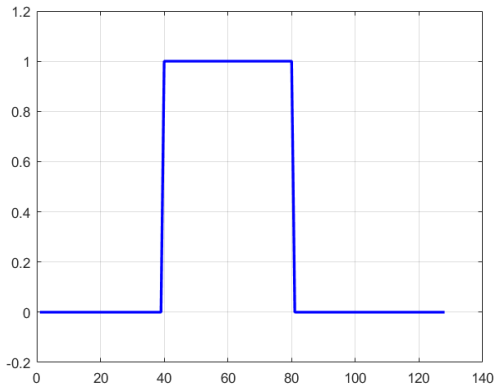


Figura 12.7: Rezultatul filtrării inverse.

```
Q = 1./H;
figure;
plot(w, abs(fftshift(Q)));

F_degr = fft(f_degr);
F_rest = F_degr .* Q;
f_rest = ifft(F_rest);
figure;
plot(f_rest);
```

12.2.2 Restaurarea prin filtrare inversă, în prezența zgomotului

Să presupunem că după filtrarea trece-jos, se suprapune un zgomot de tip Gaussian, $r(t)$, de medie zero, de tipul celui prezentat în Figura 12.8. Semnalul degradat va arăta ca cel prezentat în Figura 12.9. Se poate observa că, pe lângă modificarea formei caracteristică filtrării trece jos (modificarea pantei trecerilor bruște), zgomotul perturbă într-un mod aleator palierele semnalului.

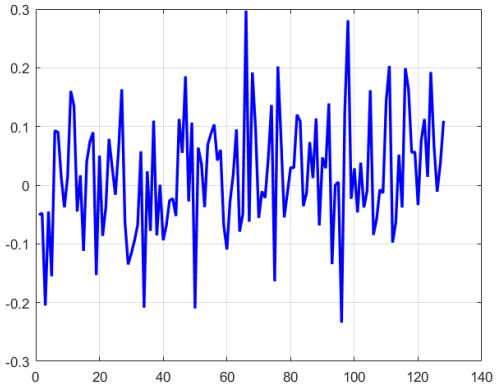


Figura 12.8: Zgomotul de tip Gaussian $r(t)$.

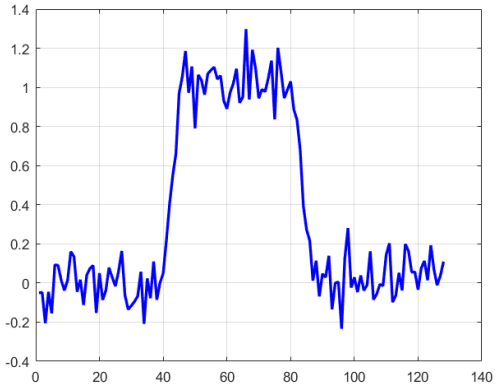
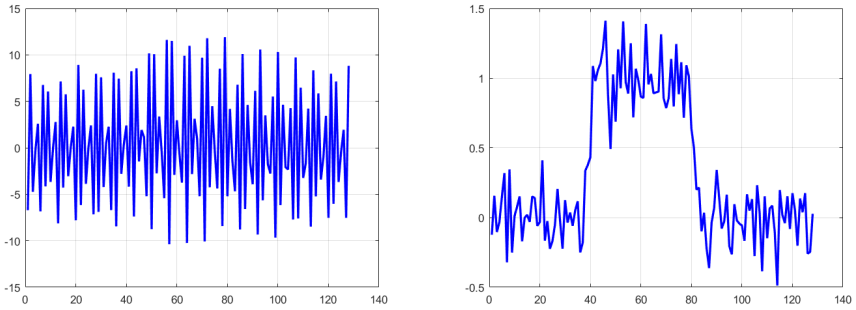


Figura 12.9: Semnalul degradat după suprapunerea zgomotului.

```
r = randn(1, N) / 10;
figure;
plot(r);
```

```
f_degr = f_degr + r;
figure;
plot(f_degr);
```

Semnalul "refăcut" prin filtrare inversă este cel prezentat în Figura 12.10(a).



(a) (b)

Figura 12.10: Semnalul refăcut: (a) fără prăguire, (b) cu prăguire.

Se poate observa cum amplificările exagerate și nedorite ale anumitor frecvențe prezente doar în zgomot, cauzate de polii funcției $Q(\omega)$, au dus la o refacere eronată, inutilizabilă. O soluție ar fi eliminarea amplificărilor exagerate din funcția $Q(\omega)$, iar cea mai simplu mod este de a prăgui această funcție cu un prag T, practic limitând amplificările nedorite.

```
T = 2;
Q_prag = Q;
Q_prag(abs(Q_prag) > T) = T;
F_degr = fft(f_degr);
F_rest = F_degr .* Q;

f_rest = ifft(F_rest);
figure;
plot(f_rest);
F_rest_prag = F_degr .* Q_prag;
f_rest_prag = ifft(F_rest_prag);
figure;
plot(f_rest_prag);
```

Se poate observa în Figura 12.10(b) că în acest al doilea caz panta semnalului original este refăcută corect, cu preţul unor variaţii în zonele uniforme ale semnalului, variaţii care sunt mult mai mici de această dată; aceste

variații pot fi eliminate printr-o filtrare adecvată (cum ar fi cea adaptivă), astfel încât și palierele să fie refăcute (la 0, respectiv 1).

12.3 Filtrul invers cu constrângeri

Filtrul invers cu constrângeri corectează efectele nedorite ale filtrului invers, luând în calcul energia E_r a zgomotului. Semnalul restaurat $\hat{f}(t)$ este calculat astfel încât energia diferenței între semnalul degradat $f'(t)$ și cel obținut prin trecerea semnalului restaurat prin filtrul de degradare $h(t)$ să fie egală cu energia zgomotului.

$$\int_{-\infty}^{+\infty} \left(\int_{-\infty}^{+\infty} h(\tau) f(t-\tau) d\tau - f''(t) \right)^2 dt = E_r \tag{12.6}$$

Trecând în domeniul spectral prin Transformata Fourier, ecuația devine:

$$\int_{-\infty}^{+\infty} \left| H(\omega)\widehat{F}(\omega) - F''(\omega) \right|^2 d\omega = E_r \tag{12.7}$$

Deoarece ecuația admite o infinitate de soluții pentru necunoscuta $\hat{F}(\omega)$, trebuie introdusă o constrângere. O constrângere uzuală o reprezintă impunerea condiției de minimizare a energiei componentelor de înaltă frecvență ale semnalului restaurat:

$$\int_{-\infty}^{+\infty} \left(\int_{-\infty}^{+\infty} c(\tau) f(t-\tau) d\tau \right)^2 \rightarrow \min \tag{12.8}$$

unde $c(t)$ reprezintă funcția pondere a unui filtru liniar trece-sus.

În domeniul spectral ecuația de mai sus devine:

$$\int_{-\infty}^{+\infty} \left| C(\omega)\widehat{F}(\omega) \right|^2 \rightarrow \min$$

Problema este una de minimizare cu constrângeri pornind de la expresia funcției $\Psi\{\widehat{F}(\omega)\}$:

$$\Psi\{\widehat{F}(\omega)\} = \int_{-\infty}^{+\infty} \left| C(\omega)\widehat{F}(\omega) \right| - \lambda \int_{-\infty}^{+\infty} \left| H(\omega)\widehat{F}(\omega) - F''(\omega) \right|^2 d\omega$$

Expresia lui $\Psi\{\widehat{F}(\omega)\}$ se derivează în funcție de $\hat{F}(\omega)$, considerată a fi o funcție complexă în cazul general de forma $\hat{F}(\omega) = A(\omega) + jB(\omega)$,

se egalează derivatele parțiale cu zero și se rezolvă ecuațiile ce rezultă din anularea derivatelor. Estimatul $\hat{F}(\omega)$ va fi:

$$\widehat{F}(\omega) = \frac{\lambda H^*(\omega) F''(\omega)}{|C(\omega)|^2 + \lambda |C(\omega)|^2} \tag{12.9}$$

unde λ este un parametru pozitiv, care se calculează din relația:

$$\int_{-\infty}^{+\infty} \frac{|F''(\omega)|^2 |C(\omega)|^4}{\left(|C(\omega)|^2 + \lambda |H(\omega)|^2\right)^2} = E_r \tag{12.10}$$

În cazul discret, relația devine:

$$\frac{1}{N} \sum_{i=1}^{N} \frac{|F''(\omega_i)|^2 |C(\omega_i)|^4}{\left(|C(\omega_i)|^2 + \lambda |H(\omega_i)|^2\right)^2} = E_r \tag{12.11}$$

unde ω reprezintă frecvențele discrete, iar N reprezintă numărul de coeficienți din spectrul Fourier al semnalului.

Expresia răspunsului în frecvență al filtrului (de restaurare) invers cu constrângeri, ca soluție la ecuația de mai sus, este următoarea:

$$Q(\omega) = \frac{\widehat{F}(\omega)}{F''(\omega)} = H^{-1}(\omega) \frac{\lambda |H(\omega)|^2}{|C(\omega)|^2 + \lambda |H(\omega)|^2} \tag{12.12}$$

Filtrul poate fi privit ca o cascadă de două filtre:

- primul, un filtru invers de restaurare (având funcția de transfer dată de $H^{-1}(\omega)$)
- al doilea, un filtru de corecție a efectelor negative ale primului

Dacă $|H(\omega)|^2 \gg |C(\omega)|^2$, al doilea factor devine ≈ 1. Dacă $|H(\omega)|^2 \ll |C(\omega)|^2$, al doilea factor devine $\approx C^{-1}$.

12.3.1 Restaurarea prin filtrare inversă cu constrângeri

Filtrul trece-sus utilizat în implementarea operației de filtrare inversă cu constrângeri poate fi, de exemplu, un derivator de ordinul unu, având funcția pondere de mai jos:

$$c(t) = [-1 \; 1] \tag{12.13}$$

Răspunsul în frecvență $C(\omega)$ al derivatorului este prezentat în Figura 12.11.

```
c = [-1 1];
C = fft(c, N);
figure;
plot(w, abs(fftshift(C)));
```

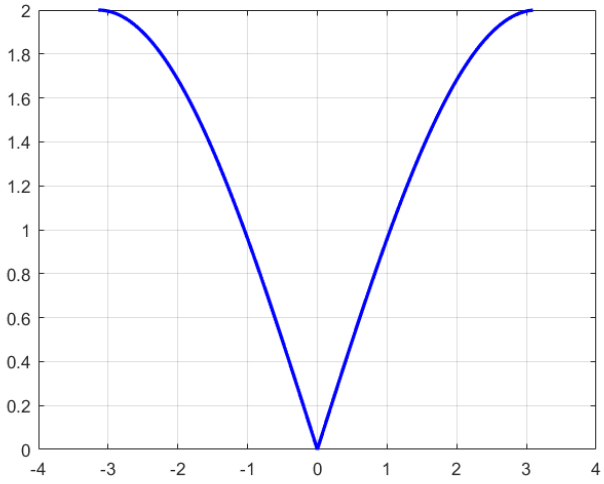


Figura 12.11: Răspunsul în frecvență $|C(\omega)|$ al derivatorului de ordinul unu.

Pentru o implementare simplă se va utiliza o valoare determinată empiric a parametrului λ:

```
lambda = 0.2;
Q_constr = 1./H .* ((lambda * abs(H) .^2 )./...
(abs(C) .^2 + lambda * abs(H) .^2));

F_rest_constr = F_degr .* Q_constr;
f_rest_constr = ifft(F_rest_constr);
figure;
plot(f_rest_constr);
```

Rezultatul filtrării este prezentat în Figura 12.12.

12.4 Filtrul Wiener

Filtrul Wiener se bazează pe o abordare statistică a operației de restaurare. Valorile semnalului restaurat sunt calculate astfel încât să minimizeze eroarea pătratică medie față de valorile semnalului original:

$$E\left\{\left(f(t)-\hat{f}(t)\right)^2\right\} \rightarrow \min \tag{12.14}$$

Dezvoltând relația de mai sus, obținem următoarea expresie pentru răspunsul în frecvență al filtrului de restaurare Wiener:

$$Q(\omega) = H^{-1}(\omega)\frac{|H(\omega)|^2\Phi_f(\omega)}{|H(\omega)|^2\Phi_f(\omega)+\Phi_r(\omega)} \tag{12.15}$$

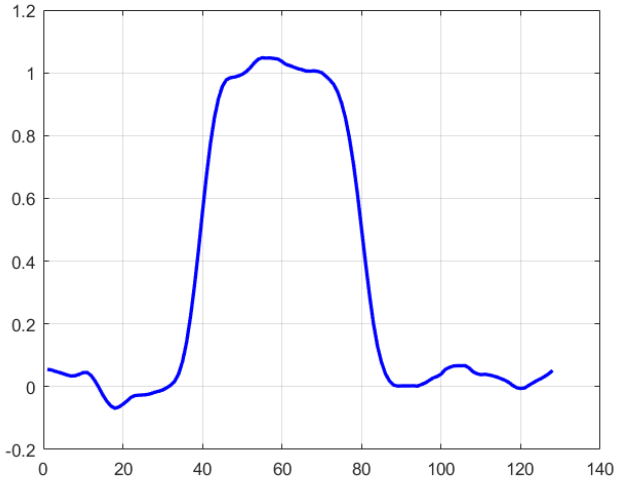


Figura 12.12: Rezultatul filtrării inverse cu constrângeri pentru semnalul din Figura 12.2.

unde $\Phi_f(\omega)$ și $\Phi_r(\omega)$ reprezintă densitatea spectrală de putere a semnalului util $f(t)$, respectiv a zgomotului $r(t)$.

Interpretarea poate fi similară cu cea a filtrului invers cu constrângeri: o cascadare a unui filtru invers de restaurare cu un filtru de corecție, care înglobează cunoștințe despre distribuția în frecvență a puterii semnalului util, respectiv a zgomotului. Densitatea spectrală de putere a semnalului util este în general necunoscută și trebuie estimată din datele disponibile: o singură realizare particulară a semnalului degradat.

O modalitate simplă, practică, dar nu foarte precisă de a estima cantitatea $\Phi_f(\omega)$ este următoarea:

$$\Phi_f(\omega) \approx |F(\omega)|^2 \approx \begin{cases} |F''(\omega)|^2 - |R(\omega)|^2 & \text{dacă} \\ & |F''(\omega)| \geq |R(\omega)| \\ 0 \quad \text{în rest} \end{cases} \tag{12.16}$$

Densitatea spectrală de putere a zgomotului este în general aproximată cu o constantă A (zgomotul fiind presupus alb):

$$\Phi_r(\omega) \approx |R(\omega)|^2 = A \tag{12.17}$$

Operația de filtrare se poate implementa în Matlab utilizând funcția `deconvwnr`, astfel:

```
f_rest_wiener = deconvwnr(f_degr, h, nsr);
figure;
plot(f_rest_wiener);
```

unde raportul zgomot-semnal `nsr` (Noise-to-signal Ratio) se estimează ca raportul dintre varianța zgomotului și cea a semnalului. Pentru utilizarea funcției în mediul Octave, este necesară încărcarea prealabilă a pachetului *image*, aceasta facându-se prin introducerea următoarei comenzi:

```
>>pkg load image
```

Rezultatul restaurării utilizând filtrul Wiener este prezentat în Figura 12.13.

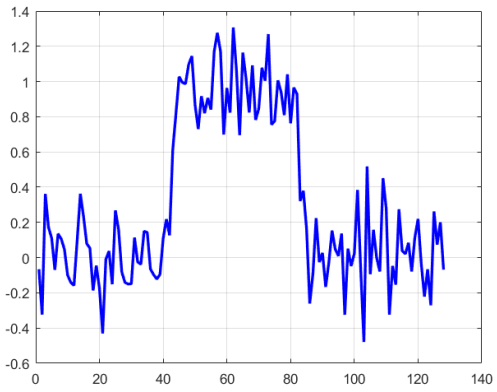


Figura 12.13: Semnalul restaurat utilizând filtrul Wiener.

12.5 Exerciții

1. Repetați pașii operației de filtrare inversă pentru diverse valori ale pragului T.

2. Modificați filtrul de mediere utilizat pentru degradarea semnalului, astfel încât să aibă o fereastră de dimensiune 5, respectiv 9.

3. Estimați energia zgomotului E_r utilizând relația (12.3), apoi determinați valoarea optimă a parametrului λ, pe baza relației (12.11).

Bibliografie selectivă

[1] Ciuc, M., Vertan, C., *Prelucrarea Statistică a Semnalelor*, Editura MatrixRom, 2005

[2] Florea, C., Florea, L., *Procesarea digitală a semnalelor – Îndrumar de laborator*, Editura MatrixRom, 2008

[3] Ingle, V.K., Proakis, J.G., *Digital signal processing using MATLAB: a problem solving companion*, Cengage Learning, 2016

[4] Kalechman, M., *Practical Matlab Basics for Engineers*, CRC Press, Boca Raton, FL, 2008

[5] Kertesz, C., Ivanovici, M., *Procesarea digitală a semnalelor – Îndrumar de laborator*, Editura Universităţii Transilvania, Braşov, 2009

[6] Mitra, S.K., Kuo, Y., *Digital signal processing: a computer-based approach*, McGraw-Hill, New York, 2006

[7] Oppenheim, A.V., Schafer, R.W., *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, New Jersey, 1975

[8] Smith, J.O., *Introduction to Matlab and Octave*, Center for Computer Research in Music and Acoustics, Stanford University, Stanford, CA, februarie 2009

[9] Smith, S.W., *The Scientist and Engineer's Guide to Digital Signal Processing*, 2 ed., California Technical Publishing, San Diego, 1999

[10] Spătaru, A., *Teoria Transmisiunii Informaţiei*, Editura Didactică şi Pedagogică, Bucureşti, 1983