Ministerul Educației și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică și Microelectronică

Departamentul Ingineria Software și Automatică

**RAPORT**

**Lucrarea de laborator nr.4**

**la Prelucrarea Semnalelor**

*Tema: Eșantionarea și cuantizarea semnalelor.*

*Interpolarea semnalelor eșantionate.*

**Grupa academică:**  TI-211

**A efectuat:**  Popa Cătălin

**A verificat:**  Potlog Mihail

Chișinău 2024

**Exercițiul 1.1**

Reprezentați grafic 1024 de eșantioane ale unui semnal alcătuit din 2 sinusoide (una cu frecvența de 50 Hz, defazajul 0 și amplitudinea 0.5 V, iar cealaltă cu frecvența de 230 Hz, defazajul π/3 și amplitudinea 0.2 V), folosind o frecvență de eșantionare de 8 kHz.

Codul sursă

% Parametrii semnalului

f1 = 50; % Frecvența primului sinus (Hz)

f2 = 230; % Frecvența celui de-al doilea sinus (Hz)

phi1 = 0; % Defazajul primului sinus

phi2 = pi/3; % Defazajul celui de-al doilea sinus

A1 = 0.5; % Amplitudinea primului sinus (V)

A2 = 0.2; % Amplitudinea celui de-al doilea sinus (V)

% Frecvența de eșantionare

Fs = 8000; % Frecvența de eșantionare (Hz)

% Timpul și eșantionarea

t = 0:1/Fs:1023/Fs; % Vector de timp (0 până la 1023 de eșantioane la o frecvență de eșantionare de 8 kHz)

% Generarea semnalului format din cele două sinusoide

x = A1 \* sin(2\*pi\*f1\*t + phi1) + A2 \* sin(2\*pi\*f2\*t + phi2);

% Reprezentarea grafică a semnalului

plot(t, x);

xlabel('Timp (s)');

ylabel('Amplitudine (V)');

title('Semnal format din două sinusoide');

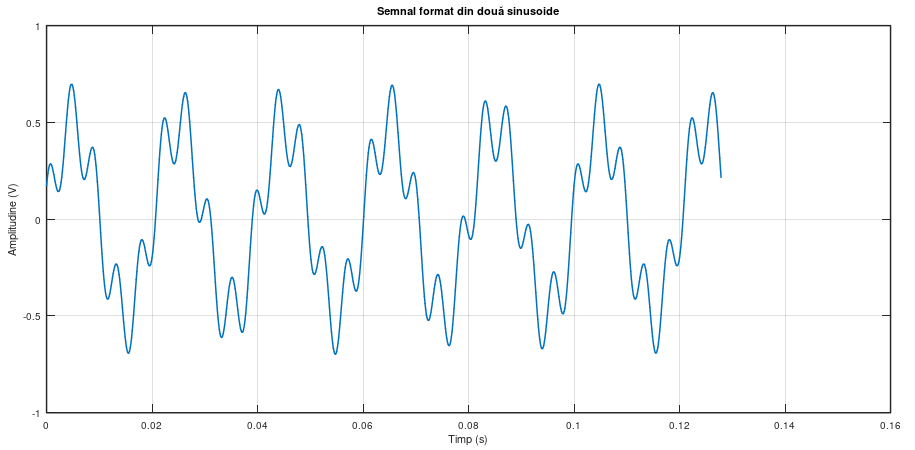
grid on;

Figura 1.1 – 1024 de eșantioane ale unui semnal alcătuit din 2 sinusoide

**Exercițiu 1.2**

Să se genereze în MATLAB semnalul din figura de mai jos. Indiciu: pornind de la figură, trebuie să se identifice toți parametrii sinusoidei (amplitudine, frecvență, frecvență de eșantionare, durată, fază inițială). Frecvența fs a fost aleasă astfel, încât să fie 30 de eșantioane într-o perioadă.

% Parametrii semnalului

A = 1; % Amplitudinea (V)

f = 1; % Frecvența (Hz)

durata = 3; % Durata (secunde)

faza = 0; % Faza inițială (rad)

fs = 30\*f; % Frecvența de eșantionare (eșantioane pe perioadă)

% Timpul și eșantionarea

t = 0:1/fs:durata-1/fs; % Vector de timp (de la 0 la durata, cu pasul 1/fs)

% Generarea semnalului sinusoidal

x = A \* sin(2\*pi\*f\*t + faza);

% Reprezentarea grafică a semnalului cu stem

stem(t, x);

xlabel('Timp (s)');

ylabel('Amplitudine (V)');

title('Semnal sinusoidal');

grid on;

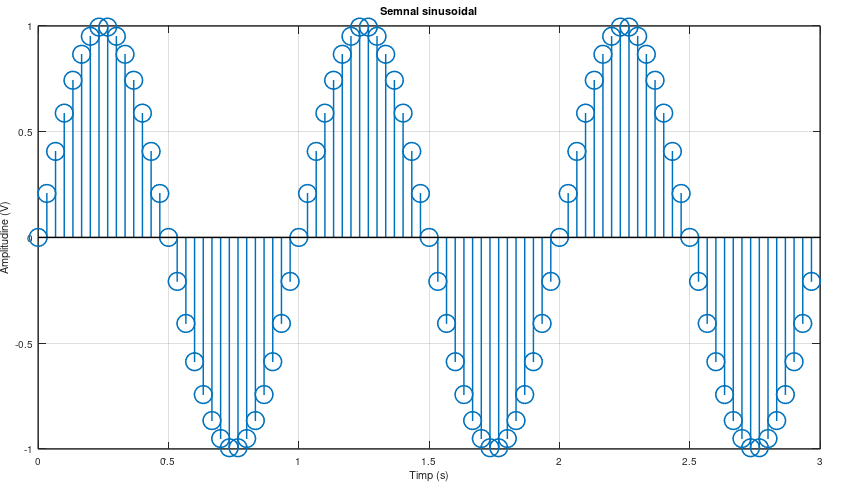


Figura 1.2 – Semnal sinusoidal

**Exercițiu 1.3**

Fie semnalul:

x(t) = 10 sin (200πt +π/2 ) + 20 sin(100πt) - 40sin (300πt -π/4 ).

Cerințe:

* care este frecvența minimă de eșantionare astfel încât să se respecte teorema eșantionării;
* alegând o frecvență de eșantionare de 10 ori mai mare decât cea determinată la punctul anterior, să se eșantioneze semnalul x(t) și să se reprezinte graphic
* care este frecvența de repetiție a semnalului x(t)

% Definirea semnalului

t = 0:0.0001:0.1; % Definirea timpului pentru grafic (0-0.1 secunde)

x = 10\*sin(200\*pi\*t + pi/2) + 20\*sin(100\*pi\*t) - 40\*sin(300\*pi\*t - pi/4);

% Frecvența de eșantionare

Fs = 6000; % Hz

% Eșantionarea semnalului

n = 0:1/Fs:0.1; % Eșantionare de la 0 la 0.1 secunde cu frecvența de eșantionare Fs

xn = 10\*sin(200\*pi\*n + pi/2) + 20\*sin(100\*pi\*n) - 40\*sin(300\*pi\*n - pi/4);

% Reprezentarea grafică a semnalului eșantionat

figure;

plot(n, xn);

xlabel('Timp (s)');

ylabel('Amplitudine');

title('Semnal eșantionat');

grid on;

% Frecvența de repetiție a semnalului

f\_rep = 100; % Hz

fprintf('Frecvența de repetiție a semnalului x(t) este: %d Hz\n', f\_rep);

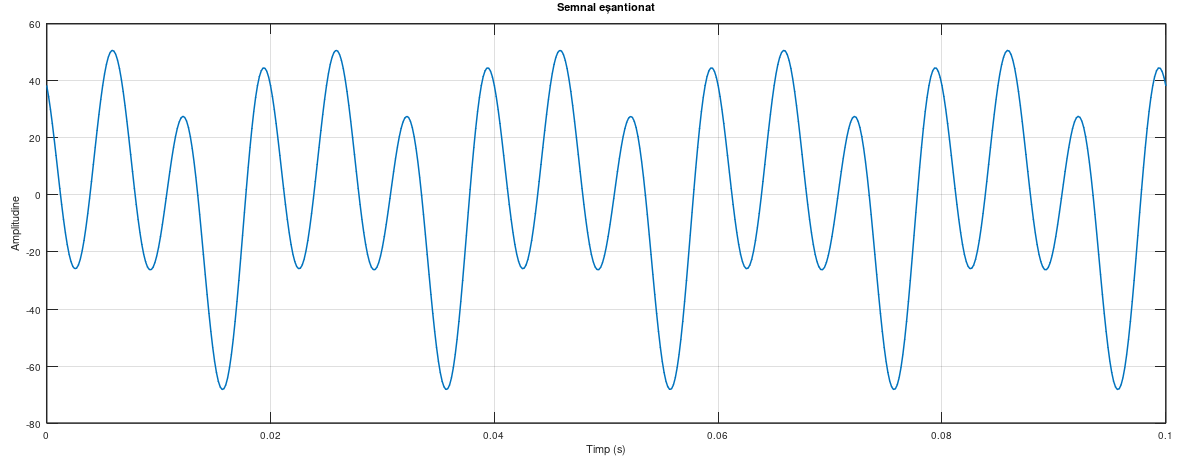


Figura 1.3 – Semnal eșantionat

**Exercițiu 1.4**

Fie semnalul sinusoidal 𝑥(𝑡) = 3sin(2𝜋 ∙ 5𝑡). Să se eșantioneze acest semnal cu fs1 = 4Hz și cu fs2 = 50Hz și să se reprezinte grafic. În care dintre cele două cazuri poate fi reconstituit semnalul x(t) din eșantioanele sale?

Pentru Fs = 4 Hz

% Definirea semnalului

t = 0:0.01:1; % Timpul de la 0 la 1 secundă cu pasul de 0.01 secunde

x = 3 \* sin(2\*pi\*5\*t);

% Frecvența de eșantionare

fs1 = 4; % Hz

% Eșantionarea semnalului

n1 = 0:1/fs1:1; % Eșantionare de la 0 la 1 secundă cu frecvența de eșantionare fs1

xn1 = 3 \* sin(2\*pi\*5\*n1);

% Reprezentarea grafică a semnalului eșantionat pentru fs1

figure;

stem(n1, xn1);

xlabel('Timp (s)');

ylabel('Amplitudine');

title('Semnal eșantionat pentru fs1 = 4 Hz');

grid on;

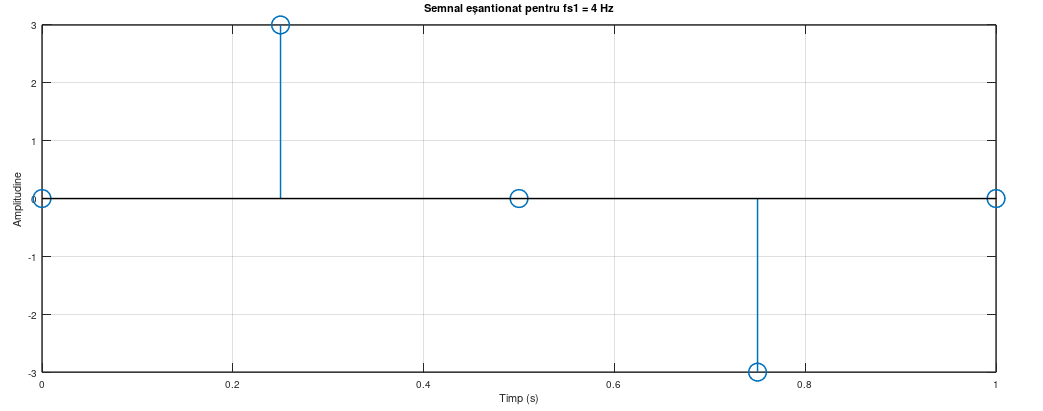


Figura 1.4.1 – Fs este 4Hz

Pentru Fs = 50 Hz

% Frecvența de eșantionare

fs2 = 50; % Hz

% Eșantionarea semnalului

n2 = 0:1/fs2:1; % Eșantionare de la 0 la 1 secundă cu frecvența de eșantionare fs2

xn2 = 3 \* sin(2\*pi\*5\*n2);

% Reprezentarea grafică a semnalului eșantionat pentru fs2

figure;

stem(n2, xn2);

xlabel('Timp (s)');

ylabel('Amplitudine');

title('Semnal eșantionat pentru fs2 = 50 Hz');

grid on;

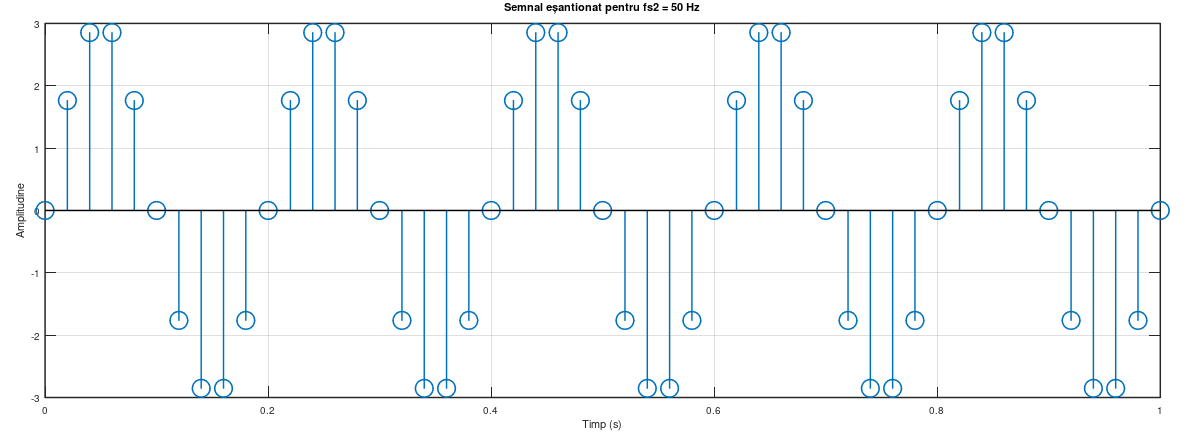
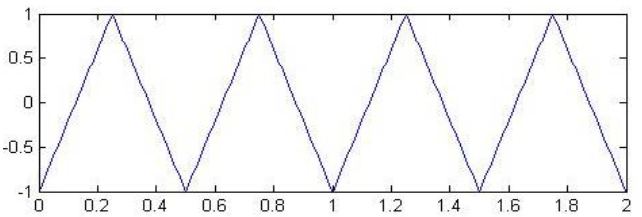


Figura 1.4.2 Fs este 50 Hz

După reprezentarea grafică a ambelor semnale eșantionate, putem observa că pentru fs1=4 Hz semnalul este sub-eșantionat și nu poate fi reconstituit corect. În schimb, pentru fs2 = 50 semnalul este eșantionat corect și poate fi reconstituit din eșantioanele sale. Deci, semnalul poate fi reconstituit din eșantioanele sale doar în cazul în care frecvența de eșantionare este mai mare decât dublul frecvenței maxime a semnalului.

**Exercițiu 1.5**

Să se genereze și să se prezinte în MATLAB, în două ferestre separate, semnalul din figura următoare și semnalul discretizat.



Indiciu:

Pornind de la figură, trebuie să se identifice toți parametrii semnalului triunghiular (amplitudine, frecvență, frecvență de eșantionare, durată). Frecvența fs se alege astfel încât să fie 30 de eșantioane într-o perioadă.

% Definirea timpului

t = 0:1/60:2; % Timpul cu un pas de 1/60 secunde

% Amplitudinea ajustată pentru a atinge 1 pe axa y

A = 1;

% Generarea semnalului triunghiular și scalarea acestuia

y = A \* sawtooth(4 \* pi \* t, 0.5);

% Plotează semnalul triunghiular continuu

subplot(2,1,1);

plot(t, y);

title('Semnal triunghiular continuu');

xlabel('Timp (s)');

ylabel('Amplitudine');

% Plotează semnalul triunghiular discretizat

subplot(2,1,2);

stem(t, y);

title('Semnal triunghiular discretizat');

xlabel('Timp (s)');

ylabel('Amplitudine');

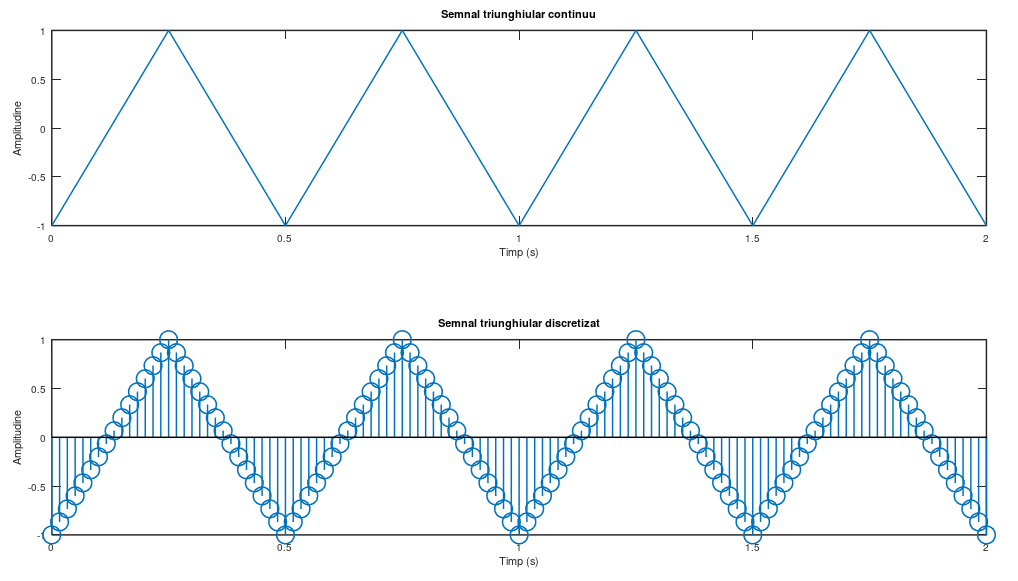


Figura 1.5 – Semnal triunghiular

**Exercițiul 2.1**

Un semnal cu durata de 2 minute este eșantionat cu frecvența de eșantionare de 4kHz. Câte eșantioane vor rezulta? Dacă fiecare eșantion este stocat pe 2 octeți, ce memorie vor ocupa toate eșantionale generate?

Numărul total de eșantioane=4000 eșantioane/sec×120sec=480 000 0eșantioane

Memorie ocupată =480000eșantioane×2octeți/eșantion=960 000octeți

**Exercițiul 2.2**

Să se cuantizeze semnalele 𝑥[𝑛] = 17/𝑛 și 𝑦[𝑛] = − 17/𝑛 , 𝑛 = 1 … 40, folosind metodele floor(trunchiere) și round(rotungire). Se cunosc nivelurile de cuantizare 1 {0, ±1, ±2, … , ±17}. Să se reprezinte grafic semnalele originale 𝑥[𝑛] și 𝑦[𝑛], precum și semnalele obținute în urma cuantizării. Să se calculeze și să se reprezinte grafic zgomotul de cuantizare.

**Generarea semnalelor originale:** Se definește un vector 𝑛 n de la 1 la 40 cu pași de 0.01 pentru a genera ținând cont de punctele de cuantizare. Se calculează semnalele 𝑥 [ 𝑛 ] x[n] și 𝑦 [ 𝑛 ] y[n] folosind expresiile date.

n = 1:0.01:40;

x = 17 ./ n;

y = -17 ./ n;

**Cuantizarea folosind metoda de rotunjire (round) și trunchiere (floor):** Se calculează semnalele cuantizate pentru 𝑥 [ 𝑛 ] x[n] și 𝑦 [ 𝑛 ] y[n] folosind metodele de rotunjire și trunchiere. Se calculează zgomotul de cuantizare 𝑧 [ 𝑛 ] = 𝑥 [ 𝑛 ] − 𝑥 𝑞 [ 𝑛 ] z[n]=x[n]−x q ​ [n] și 𝑧 [ 𝑛 ] = 𝑦 [ 𝑛 ] − 𝑦 𝑞 [ 𝑛 ] z[n]=y[n]−y q ​ [n], unde 𝑥 𝑞 [ 𝑛 ] x q ​ [n] și 𝑦 𝑞 [ 𝑛 ] y q ​ [n] reprezintă semnalele cuantizate.

**Figura 1**

**Subplotul 1 :** Afișează semnalul original 𝑥 [ 𝑛 ] x[n] (linia solidă) și semnalul cuantizat utilizând metoda de rotunjire (round) 𝑥 𝑞 [ 𝑛 ] x q ​ [n] (linia punctată). Pe axa x avem indicele de eșantionare 𝑛 n, iar pe axa y avem valoarea semnalului.

figure(1)

subplot(2,1,1)

x\_round = round(x);

plot(n, x, n, x\_round)

xlabel('n'); ylabel('x[n]'); title('Semnalul cuantizat (metoda round)');

legend('Semnal original', 'Semnal cuantizat (round)')

**Subplotul 2**: Afișează zgomotul de cuantizare 𝑧 [ 𝑛 ] = 𝑥 [ 𝑛 ] − 𝑥 𝑞 [ 𝑛 ] z[n]=x[n]−x q ​ [n] asociat metodei de rotunjire. Pe axa x avem indicele de eșantionare 𝑛 n, iar pe axa y avem valoarea zgomotului.

subplot(2,1,2)

z\_round = x - x\_round;

plot(n, z\_round)

xlabel('n'); ylabel('x[n]-xq[n]'); title('Zgomot (metoda round)');

**Figura 2**

**Subplotul 1**: Afișează semnalul original 𝑥 [ 𝑛 ] x[n] (linia solidă) și semnalul cuantizat utilizând metoda de trunchiere (floor) 𝑥 𝑞 [ 𝑛 ] x q ​ [n] (linia punctată). Pe axa x avem indicele de eșantionare 𝑛 n, iar pe axa y avem valoarea semnalului.

figure(2)

subplot(2,1,1)

x\_floor = floor(x);

plot(n, x, n, x\_floor)

xlabel('n'); ylabel('x[n]'); title('Semnalul cuantizat (metoda floor)');

legend('Semnal original', 'Semnal cuantizat (floor)')

**Subplotul 2**: Afișează zgomotul de cuantizare 𝑧 [ 𝑛 ] = 𝑥 [ 𝑛 ] − 𝑥 𝑞 [ 𝑛 ] z[n]=x[n]−x q ​ [n] asociat metodei de trunchiere. Pe axa x avem indicele de eșantionare 𝑛 n, iar pe axa y avem valoarea zgomotului.

subplot(2,1,2)

z\_floor = x - x\_floor;

plot(n, z\_floor)

xlabel('n'); ylabel('x[n]-xq[n]'); title('Zgomot (metoda floor)');

**Figura 3**

**Subplotul 1:** Afișează semnalul original 𝑦 [ 𝑛 ] y[n] (linia solidă) și semnalul cuantizat utilizând metoda de rotunjire (round) 𝑦 𝑞 [ 𝑛 ] y q ​ [n] (linia punctată). Pe axa x avem indicele de eșantionare 𝑛 n, iar pe axa y avem valoarea semnalului.

figure(3)

subplot(2,1,1)

y\_round = round(y);

plot(n, y, n, y\_round)

xlabel('n'); ylabel('y[n]'); title('Semnalul cuantizat (metoda round)');

legend('Semnal original', 'Semnal cuantizat (round)')

**Subplotul 2**: Afișează zgomotul de cuantizare 𝑧 [ 𝑛 ] = 𝑦 [ 𝑛 ] − 𝑦 𝑞 [ 𝑛 ] z[n]=y[n]−y q ​ [n] asociat metodei de rotunjire. Pe axa x avem indicele de eșantionare 𝑛 n, iar pe axa y avem valoarea zgomotului.

subplot(2,1,2)

z\_round = y - y\_round;

plot(n, z\_round)

xlabel('n'); ylabel('y[n]-yq[n]'); title('Zgomot (metoda round)');

**Figura 4**

**Subplotul 1:** Afișează semnalul original 𝑦 [ 𝑛 ] y[n] (linia solidă) și semnalul cuantizat utilizând metoda de trunchiere (floor) 𝑦 𝑞 [ 𝑛 ] y q ​ [n] (linia punctată). Pe axa x avem indicele de eșantionare 𝑛 n, iar pe axa y avem valoarea semnalului.

figure(4)

subplot(2,1,1)

y\_floor = floor(y);

plot(n, y, n, y\_floor)

xlabel('n'); ylabel('y[n]'); title('Semnalul cuantizat (metoda floor)');

legend('Semnal original', 'Semnal cuantizat (floor)')

**Subplotul 2:** Afișează zgomotul de cuantizare 𝑧 [ 𝑛 ] = 𝑦 [ 𝑛 ] − 𝑦 𝑞 [ 𝑛 ] z[n]=y[n]−y q ​ [n] asociat metodei de trunchiere. Pe axa x avem indicele de eșantionare 𝑛 n, iar pe axa y avem valoarea zgomotului.

subplot(2,1,2)

z\_floor = y - y\_floor;

plot(n, z\_floor)

xlabel('n'); ylabel('y[n]-yq[n]'); title('Zgomot (metoda floor)');

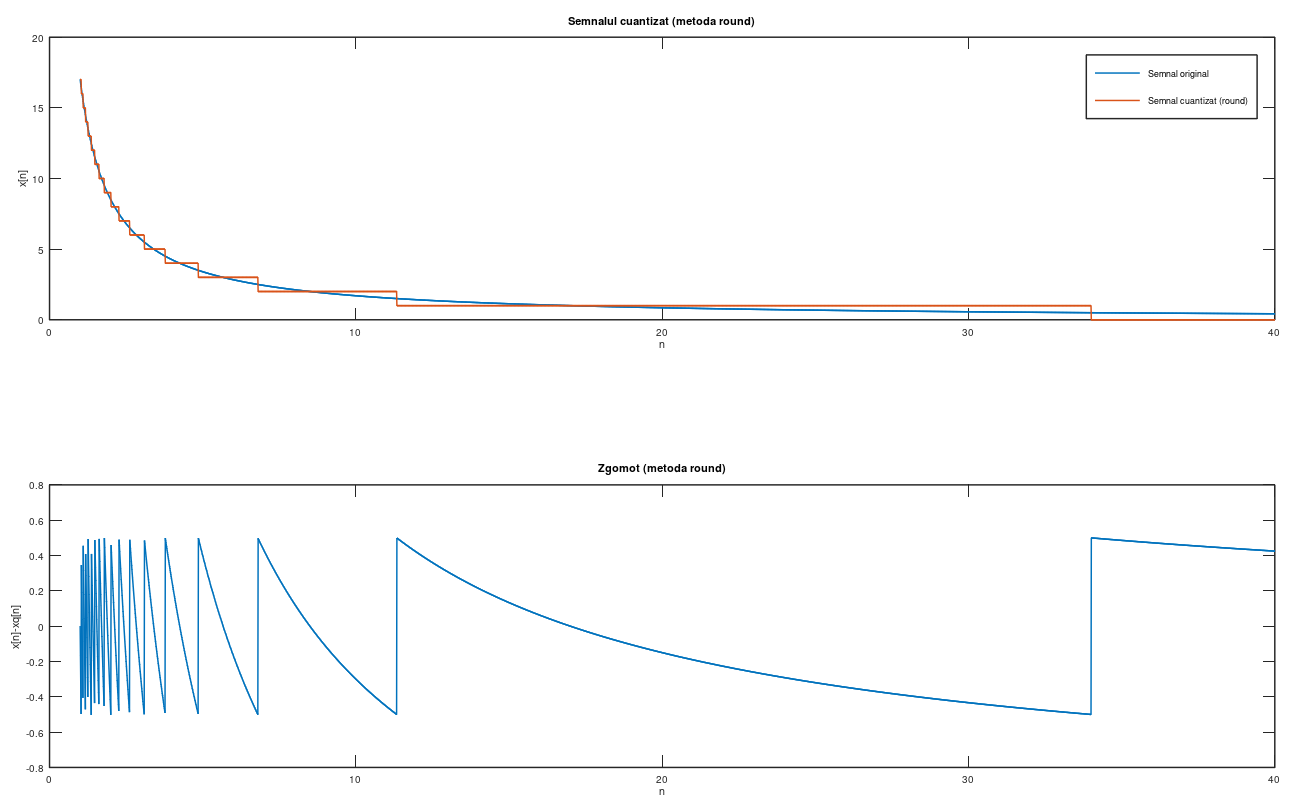


Figura 2.1 – Semnalul cuantizat (metoda round) și zgomot

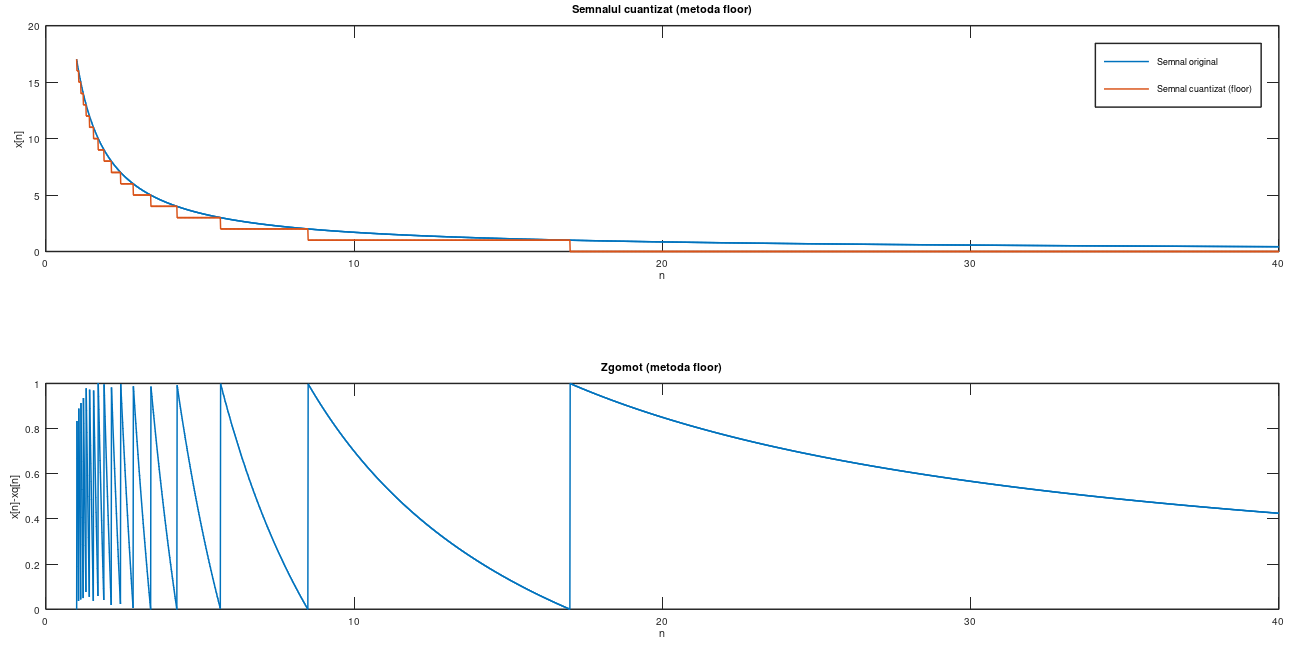


Figura 2.2 – Semnalul cuatizat (metoda floor) și zgomot

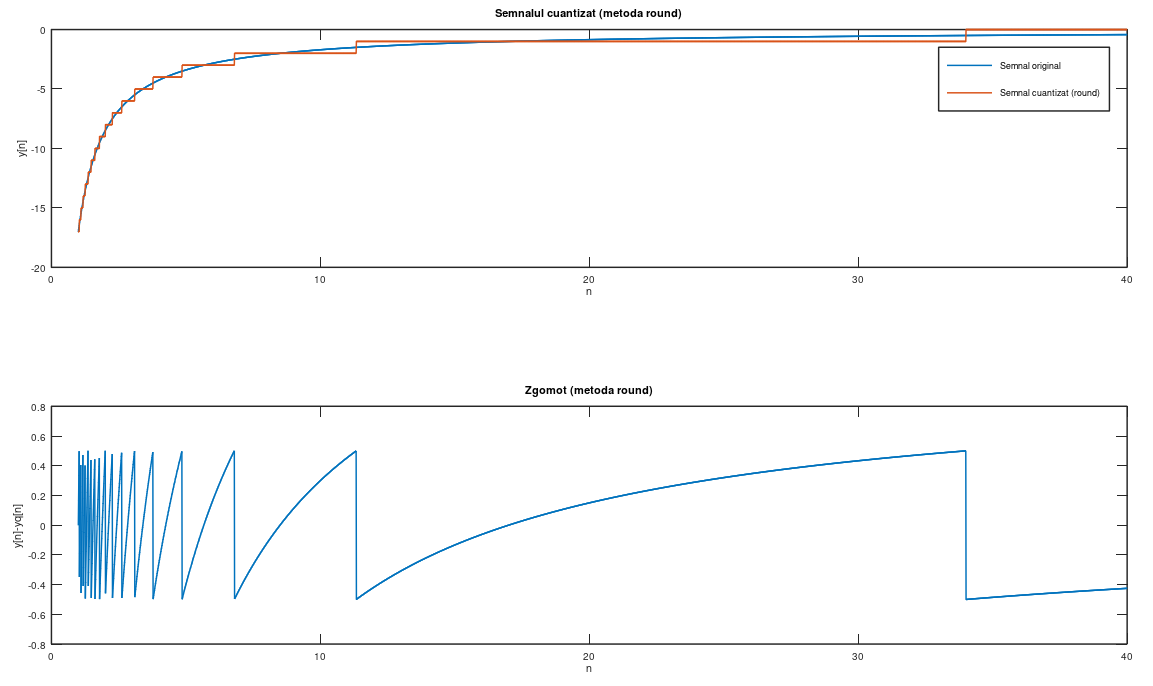


Figura 2.3 – Semnalul cuatizat (metoda round)

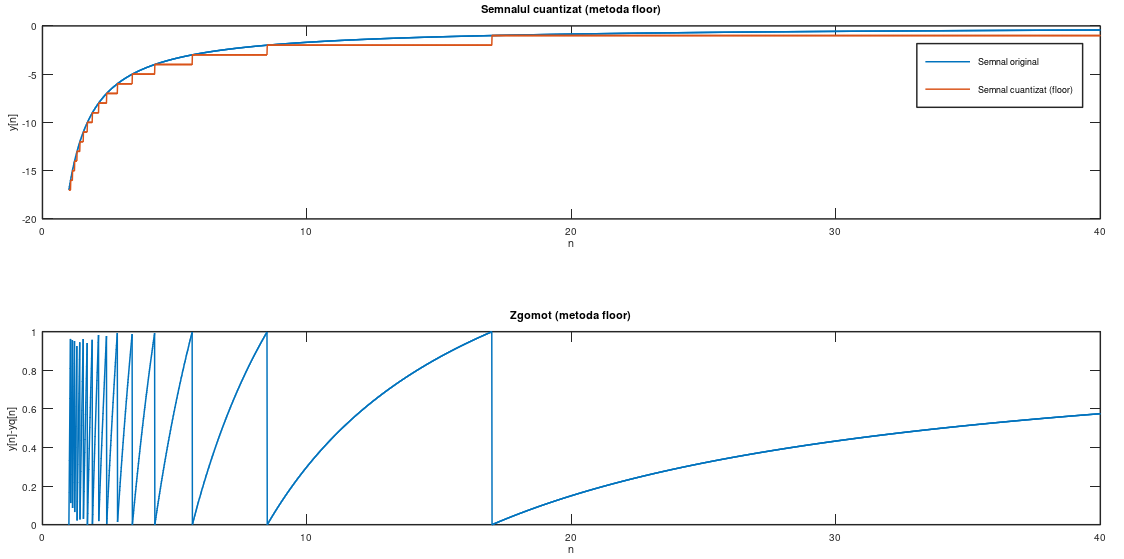


Figura 2.4 – Semnalul cuatizat (metoda floor)

**Exercițiul 2.3**

Să se genereze 15 eșantioane de zgomot alb gaussian cu media zero și dispersia 0,2.

Indiciu: pentru a genera zgomot alb gaussian se poate folosi funcția rand:

z = 0.2\*randn(1,15)

Să se cuantizeze semnalul z folosind un cuantizor uniform cu nivelurile {0, ± 1/4 , ± 2/4 , ± 3/4 , ±1}.

% generarea zgomotului alb gaussian

z = 0.2\*randn(1,15); % se genereaza 15 esantioane de zgomot alb gaussian cu media zero și dispersia 0,2 folosind functia randn

% afisarea semnalului original

plot(z);

title('Semnalul original');

figure(2)

% Se setează nivelul de cuantizare 𝑞 = ± 1 / 4 q=±1/4 și se efectuează cuantizarea semnalului utilizând metoda de rotunjire (round). Apoi, semnalul cuantizat este afișat în primul subplot.

q=0.25;

zq = round(z/q)\*q;

subplot(2,2,1), plot(zq);

title('Semnalul cuantizat (q=+-1/4)');

% Se setează nivelul de cuantizare 𝑞 = ± 2 / 4 = ± 1 / 2 q=±2/4=±1/2 și se efectuează cuantizarea semnalului utilizând metoda de rotunjire (round). Apoi, semnalul cuantizat este afișat în al doilea subplot.

q=0.5;

zq = round(z/q)\*q;

subplot(2,2,2), plot(zq);

title('Semnalul cuantizat (q=+-2/4)');

% Se setează nivelul de cuantizare 𝑞 = ± 3 / 4 q=±3/4 și se efectuează cuantizarea semnalului utilizând metoda de rotunjire (round). Apoi, semnalul cuantizat este afișat în al treilea subplot.

q=0.75;

zq = round(z/q)\*q;

subplot(2,2,3), plot(zq);

title('Semnalul cuantizat (q=+-3/4)');

% Se setează nivelul de cuantizare 𝑞 = ± 1 q=±1 și se efectuează cuantizarea semnalului utilizând metoda de rotunjire (round). Apoi, semnalul cuantizat este afișat în al patrulea subplot.

zq = round(z);

subplot(2,2,4), plot(zq);

title('Semnalul cuantizat (q=+-1)');

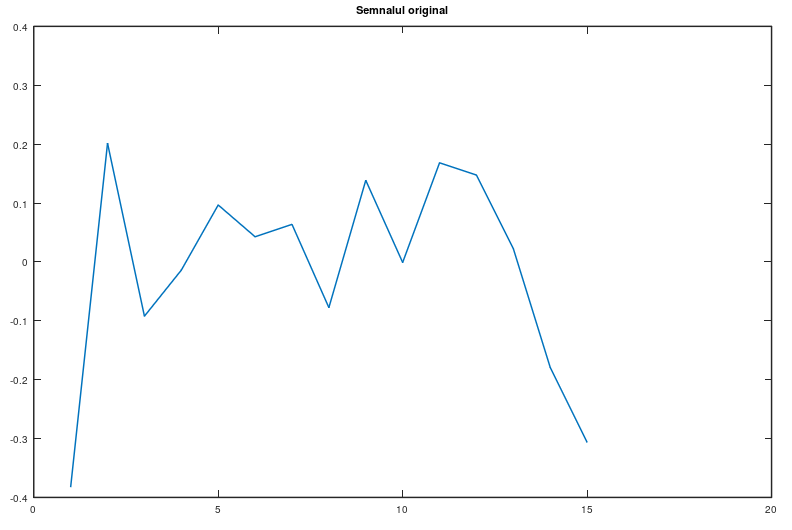


Figura 2.3.1 – Semnalul original

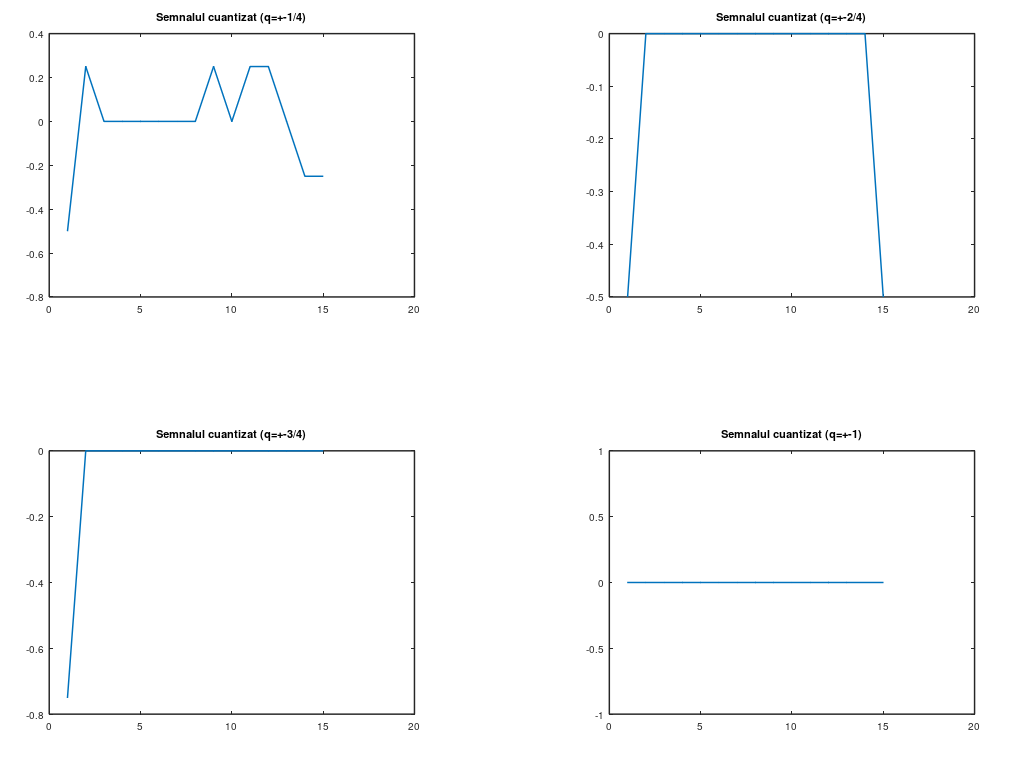


Figura 2.3.2 – Semnalul cuantizat

**Exercițiul 2.4**

Cuantizați semnalul din Exercițiul 4.1.1 pe 8 respectiv, 16 biți. Reprezentați semnalul cuantizat alături de zgomotul de cuantizare.

Se definește frecvența de eșantionare fs și vectorul timp t corespunzător. Se definește semnalul x care este compus din două componente sinusoidale cu frecvențele de 50 Hz și 230 Hz.

fs = 8000;

t = 0 : 1/fs : 1024/fs;

x = 0.5 \* sin(2 \* pi \* 50 \* t) + 0.2 \* sin(2 \* pi \* 230 \* t + pi/3);

figure;

Se definește numărul de niveluri de cuantizare N pentru 8 biți (256 niveluri). Se efectuează cuantizarea semnalului x la 8 biți folosind metoda de trunchiere (floor). Semnalul original și semnalul cuantizat pe 8 biți sunt reprezentați grafic în primul subplot.

% Cuantizare pe 8 biți

N = 256; % 8 biti

x1 = floor((N - 1) \* x) / (N - 1);

subplot(2,1,1), plot(t, x, t, x1);

xlabel('t(s)'); ylabel('A(V)'); title('Semnalul cuantizat pe 8 biti (metoda floor)');

Se definește numărul de niveluri de cuantizare N pentru 16 biți (65536 niveluri). Se efectuează cuantizarea semnalului x la 16 biți folosind metoda de trunchiere (floor). Semnalul original și semnalul cuantizat pe 16 biți sunt reprezentați grafic în al doilea subplot.

% Cuantizare pe 16 biți

N = 65536; % 16 biti

x1 = floor((N - 1) \* x) / (N - 1);

subplot(2,1,2), plot(t, x, t, x1);

xlabel('t(s)'); ylabel('A(V)'); title('Semnalul cuantizat pe 16 biti (metoda floor)');

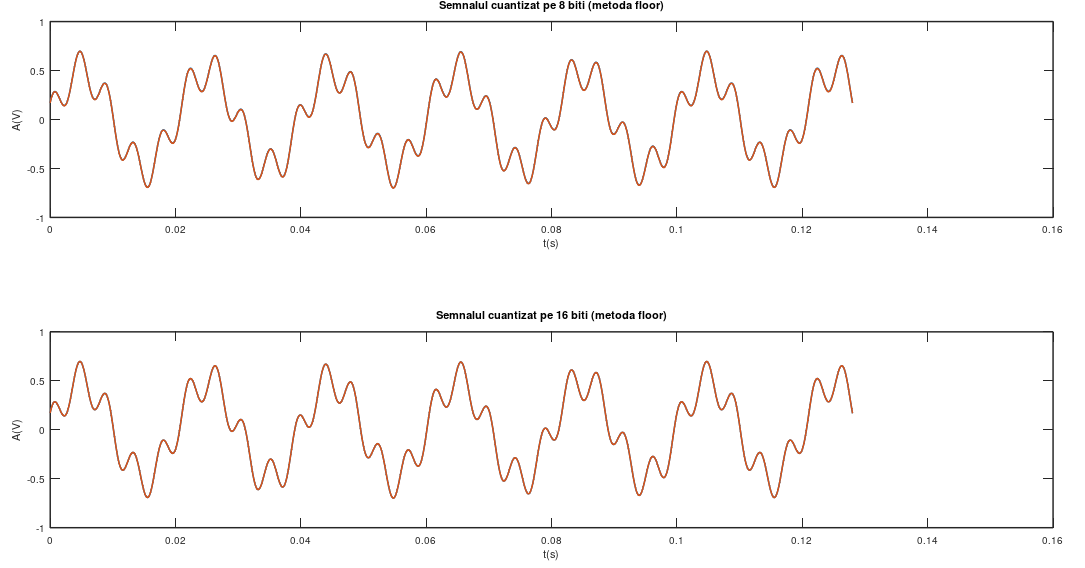


Figura 2.4 – Semnalul cuantizat pe 8 biti si 16 biti

**Exercițiul 3**

a) Executarea calculelor cu valorile recomandate în antetul script-ului, reprezentarea semnalelor în formă originară şi după reconstrucție, precum şi a diferenţelor dintre ele, în

spaţii grafice diferite sau în acelaşi spaţiu.

b) Executarea de calcule cu valorile diferite pentru frecvenţele 1f şi 2f , eventual pentru amplitudini diferite ale sinusoidelor componente, utilizând pentru fiecare caz trei valori ale frecvenţei de eşantionare: mai mică, egală şi mai mare (de 10 ori) ca frecvenţa Nyquist.

c) Aprecierea vizuală şi/sau cantitativ a modificărilor semnalului reconstituit, calitatea reconstrucției semnalului în zona de început şi de final a intervalului de timp observat.

d) Studierea şi a altor semnale de bandă limitată, recomandate de profesor.

% optiune de reprezentare (rep=0) grafic unic, (rep>0) grafice separate

tmax = 2; % timpul maxim de reprezentare

fi = pi/2; % faza componentei secundare

f1 = 8; % frecventa componentei primare

f2 = 13; % frecventa componentei secundare

pasmic = 0.001;

fes = 4.1; % frecventa de esantionare

rep = 1;

tes = 1/fes; % perioada esantioanelor

**Generarea semnalului original**

Se definește vectorul timp t cu un pas de esantionare pasmic. Se generează semnalul original y compus din două componente sinusoidale, cu frecvențele f1 și f2, respectiv.

a1 = 2.4;

a2 = 1.3;

t = 0:pasmic:tmax;

y = a1 \* cos(2 \* pi \* f1 \* t) + a2 \* cos(2 \* pi \* f2 \* t - fi);

**Eșantionarea semnalului original**

Se definește vectorul timp esantionat t1 cu un pas de esantionare tes. Se calculează numărul de esantioane n pe baza timpului total și a perioadei de esantionare. Se esantionează semnalul original y la frecvența de esantionare fes, rezultând semnalul esantionat y1.

t1 = 0:tes:tmax;

n = round(tmax/tes) + 1;

% pregatirea graficului 2 (esantionarea)

y1 = a1 \* cos(2 \* pi \* f1 \* t1) + a2 \* cos(2 \* pi \* f2 \* t1 - fi);

**Reconstruirea semnalului original**

Se reconstituie semnalul original y din semnalul esantionat y1 folosind funcția sinc pentru interpolare.

y2 = y1(1) \* sinc(t/tes);

for k = 1:(n - 1)

y2 = y2 + y1(k + 1) \* sinc(t/tes - k);

end

**Afișarea graficilor**

Dacă variabila rep este mai mare decât zero, se afișează fiecare etapă în subploturi separate. Fiecare subplot reprezintă: semnalul original, semnalul esantionat, semnalul reconstituit și diferența dintre semnalul original și cel reconstituit. Dacă rep este zero, toate graficele sunt reprezentate în același grafic.

if rep > 0

subplot(4,1,1)

plot(t,y) % trasarea graficului 1 (semnal)

hold on

ylabel('Original')

title('ESANTIONAREA SEMNALELOR')

plot([0 tmax],[0 0])

for i = 1:n

subplot(4,1,2)

% trasarea graficului 2 (esantionarea)

plot([t1(i) t1(i)],[0 y1(i)],'b:')

hold on

ylabel('Esantionat')

plot([0 tmax],[0 0])

end

subplot(4,1,3)

% trasarea graficului 3 (reconstituire)

plot(t,y2,'k')

hold on

plot([0 tmax],[0 0])

for i = 1:n

plot([t1(i) t1(i)],[0 y1(i)],'b:')

end

ylabel('Reconstituire')

subplot(4,1,4)

% trasarea graficului 4 (diferente)

plot(t,y-y2,'r')

hold on

plot([0 tmax],[0 0])

ylabel('Diferente')

xlabel('Timp(s)')

else

plot(t,y) % trasarea graficului 1 (semnal)

hold on

plot([0 tmax],[0 0])

for i = 1:n

% trasarea graficului 3 (reconstituire)

plot([t1(i) t1(i)],[0 y1(i)],'b:')

end

plot(t,y2,'k')

% trasarea graficului 4 (diferente)

plot(t,y-y2,'r')

plot([0 tmax],[0 0])

title('ESANTIONAREA SEMNALELOR')

ylabel('Semnal(albastru)/Reconstituire(negru)/Diferente(rosu)')

xlabel('Timp(s)')

end

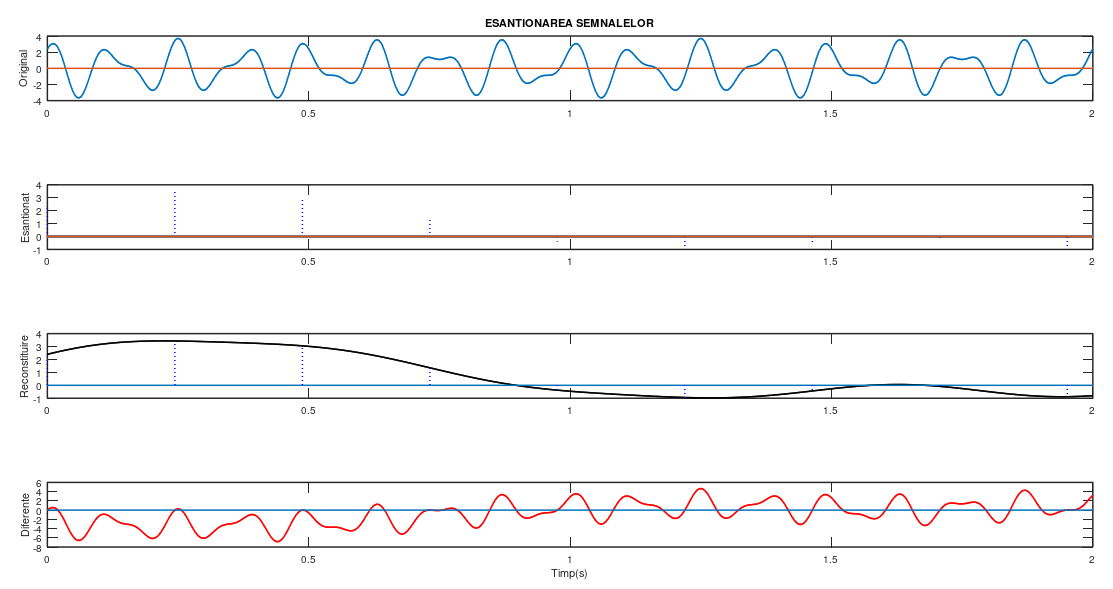


Figura 3. 1 – Fes = 4.1

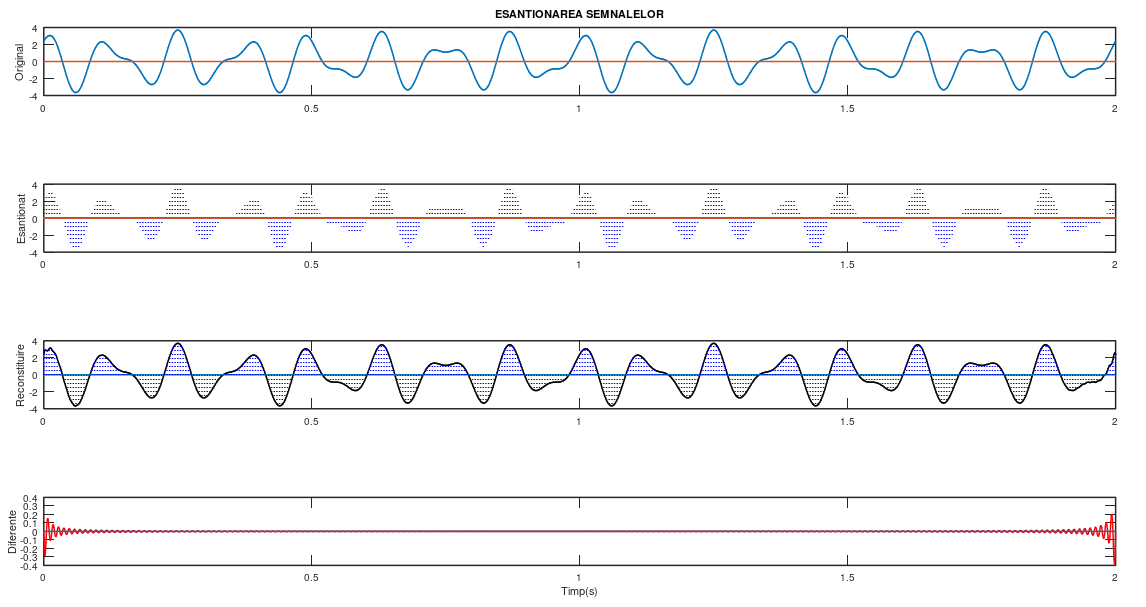


Figura 3.2 – Fes = 200

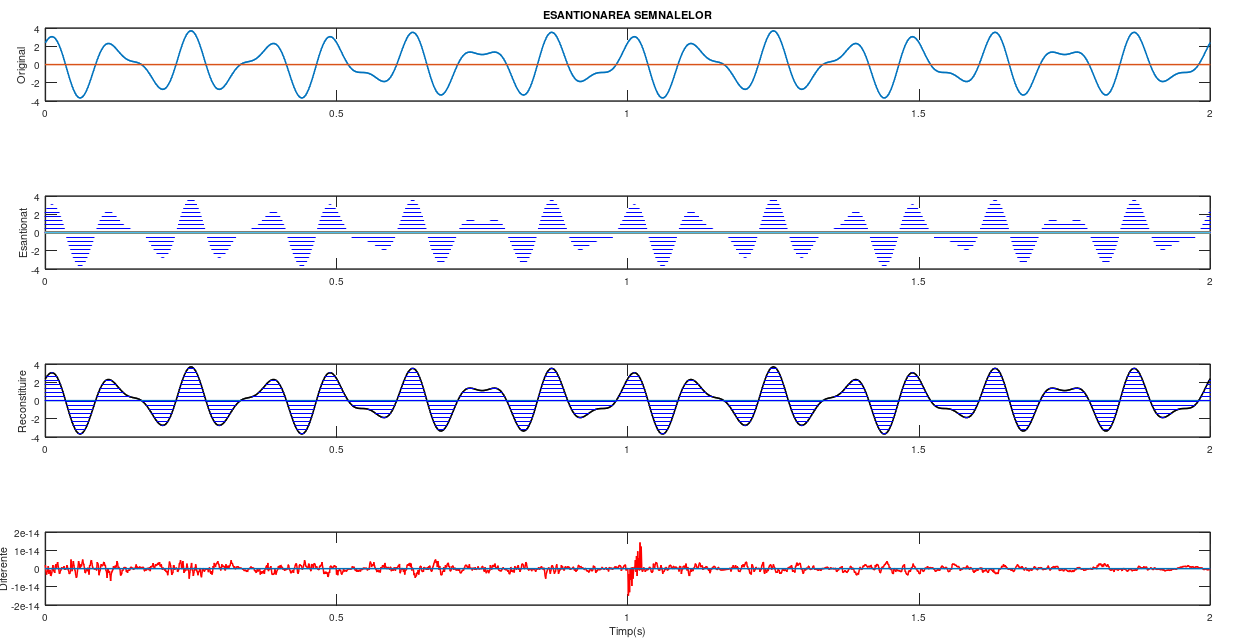


Figura 3.3 – Fes = 1000

**Concluzii**

Lucrarea a evidențiat importanța alegerii adecvate a parametrilor în prelucrarea semnalelor, începând de la frecvența de eșantionare până la metoda de cuantizare. Respectarea teoremei eșantionării și a principiilor de cuantizare au fost fundamentale pentru obținerea unei reprezentări precise a semnalelor. Reconstrucția semnalului din eșantioanele sale a demonstrat necesitatea unei frecvențe de eșantionare suficient de mari pentru a păstra informația semnalului original. În final, analiza diferențelor între semnalul original și cel reconstruit a subliniat importanța alegerii corecte a tuturor parametrilor în procesul de prelucrare a semnalelor.