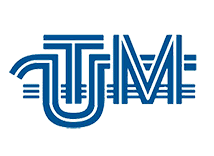
Ministerul Educaţiei, Culturii și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Ingineria Software și Automatică



**RAPORT**

Lucrare de laborator Nr.2

Disciplina: Prelucrarea semnalelor

Tema: Formarea semnalelor impuls unitare

A efectuat:

st.gr.TI-201FR

Dascal Dumitru

A verificat :

Romanenko Alexandru

conf. univ., dr.

Chișinău 2023

**1.6 Formarea semnalelor impuls unitare**

În pachetul SPTB sunt prevăzute câteva proceduri, care formează consecutivitatea datelor, care prezintă nişte semnale impuls unitare de forme tipice. Impulsul unitar de formă dreptunghiulară se poate de format cu ajutorul procedurii rectpuls, de tipul: y=rcctpuls(t,w) - care permite de a forma vectorul y a valorilor semnalului a aşa impuls de amplitudine unitate, cu lăţimea w, centrat faţă de t=0 după vectorul dat t a momentelor de timp. Dacă lăţimea impulsului w nu este dată, atunci valoarea ei se ia egală cu 1. În fig.l este prezentat rezultatul procesului, care constă din trei impulsuri dreptunghiulare consecutive de diferite înălţimi şi lăţimi, după consecutivitatea comenzilor:

>> t=0:0.01:10;

>>y=0.75\*rectpuls(t-3.2)+l.4\*rectpuls(t-5.1)+0.5\*rectpuls(t-8.04);

>> plot(t,y);

>>grid;

>> xlabel('timpul(s)');

>> ylabel('procesul de iesire y(t)')

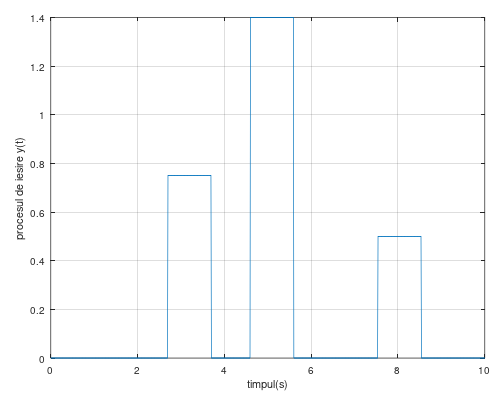


Figura 1.1 Construirea a trei impulsuri dreptunghiulare.

Impulsurile de formă triunghiulară cu amplitudine unitate se pot realiza cu ajutorul procedurii tripuls, care are forma:

y=tripuls(i,w,s).

Argumentele y, t, w, au acelaşi sens ca şi mai sus. Argumentul S (-1≤S≤1) determină înclinarea triunghiului. Dacă S=0, sau dacă nu este indicat, atunci impulsul triunghiular este simetric.

Funcţia y=tripuls(t,w) formează un impuls simetric de amplitudine unitate w centrat faţă de t=0.

>>t=0:0.01:10;

>>y=0.8\*tripuls(t-1,0.5)+1.4\*tripuls(t-3,0.8,1)+0.5\*tripuls(t-5,0.5,-1);

>>plot(t,y);

>>grid;

>>xlabel('timpul(s)');

>>ylabel('y(t)');

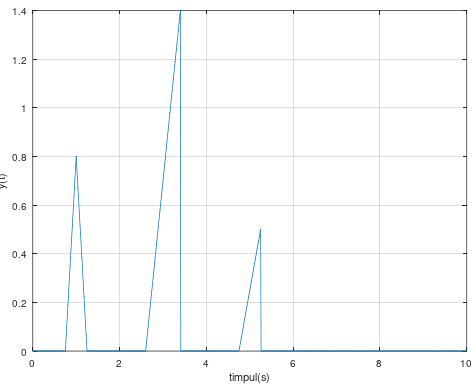


Figura 1.2 Impulsuri triunghiulare.

Formarea impulsului care este sinusoida modulată de funcţia Gauss se execută cu procedura gauspuls, adresarea către ea are forma:

yi=gauspuls(t,fc,bw)

yi=gauspuls(t,fc,bw,bwr)

|yi,yq|=gauspuls(...)

|yi,yq,ye|=gauspuls(...)

tc=gauspuIs('cutoff',fc,bw,bwr,tpe)

Funcţia yi=gauspuls(t,fc,bw) formează secvenţa eşantioanelor semnalului, calculate în momentele de timp date în vectorul t; fc determină frecvenţa sinusoidei; bw - lărgimea relativă a benzii de frecvenţă a semnalului. De exemplu, se poate lua fc= 10001Hz şi bw=0.5 (bw=∆f/fc).

Funcţia yi=gauspuls(t,fc,bw,bwr) calculează secvenţa eşantioanelor semnalului cu următorii parametri: amplitudinea=l, lăţimea benzii de frecvenţă=100bw. Graniţele benzii de frecvenţă se determină de nivelul de atenuare bwr(dB) în raport cu amplitudinea normată a semnalului. Parametrul bwr trebuie să fie negativ, de exemplu bwr=-6dB.

Funcţia [yi,yq]=gauspuls(...) calculează doi vectori. Vectorul yi conţine eşantioanele semnalului iniţial, iar vectorul yq conţine eşantioanele semnalului în care faza sinusoidei este schimbată cu 90 de grade.

Funcţia [yi,yq,ye]=gauspuls(...) calculează adăugător semnalul ye=exp(-at2) - anvelopa impulsului.

Funcţia tc=gauspuls('cutoff,fc,bw,bwr,tpe) calculează timpul tc, corespunzător momentului de timp în care amplitudinea semnalului scade pînă la tpe dB. Mărimea tpe trebuie să fie negativă, de exemplu tpe=-60dB.

**Comenzile Matlab realizate:**

**Program 1\_3**

>>t=0:0.01:10;

>> yi=0.8\*gauspuls(t-4,1,0.5);

>> plot (t,yi);

>> grid;

>> xlabel('timpul(s)');

>> ylabel('yi(t)');

Procedura sinc permite de a calcula valoarea vectorului funcţiei sinc(t), care se determină de formulele:



care prezintă o transformare inversă Fourier a impulsului dreptunghiular cu înălţimea 1 şi lăţimea 2



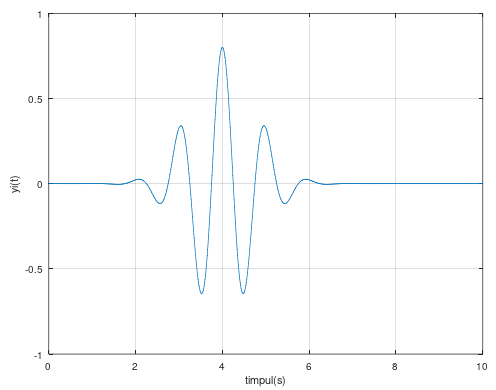


Figura 1.3 Impuls Gaussian.

**Program 1\_4**

>> t=0:0.01:50;

>> yi=0.9\*sinc(pi\*(t-25)/5);

>> plot(t,yi);

>> grid;

>> xlabel('timpul(s)')

>> ylabel('yi(t)')

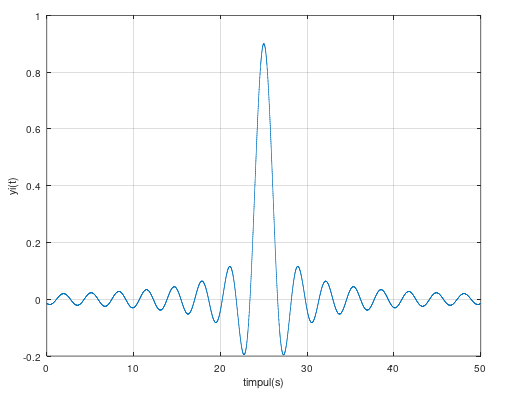


Figura 1.4 Transformata Fourier a impulsului dreptunghiular.

Funcţia y=square(t,duty) formează o secvenţă de impulsuri cu durata semiundei pozitive, care se determină de parametrul duty în procente de la perioadă. De obicei se ia duty=50. Să dăm un exemplu de utilizare a acestei proceduri (Figura 1.5).

**Program 1\_5**

>>t=-10:0.1:10;

>>y=square(t,10);

>> plot(t, y);

>> axis([min(t), max(t),-2,2]);

>> grid

În caz de probleme cu biblioteca instalată, putem exersa alt fel de cod.

>> f = 0.5;

>> y = sign(sin(2 \* pi \* f \* t));

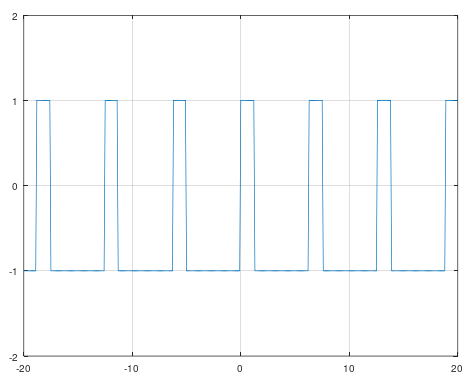


Figura 1.5 Secvenţă de impulsuri rectangulare.

Formarea semnalelor "dinte de ferestrău" şi triunghiulare cu amplitudinea ±1 şi perioada 2π se face cu procedura sawtooth:x=sawtooth(t)x=sawtooth(t,width)

Funcţia x=sawtooth(t) formează semnal de forma "dinte de ferestrău", care ia valoarea -l în momente multiple cu 2π şi crescînd liniar pe intervalul 2π cu panta 1/π.

Funcţia x=sawtooth(t,width) formează semnalul "dinte de ferestrău" modificat. Parametrul width se dă în diapazonul de la 0 la 1 şi determină o parte a perioadei, în care semnalul creşte. Semnalul creşte de la -l pînă la 1, pe intervalul de la 0 pînă la 2π\*width, iar pe urmă scade de la 1 pînă la -l pe intervalul de la 2π\*width pînă la 2π. Dacă width=0.5, atunci se formează o undă simetrică. Funcţia sawtooth(t,1) este echivalentă cu funcţia sawtooth(t). Aducem un exemplu de generare a unui semnal de tip „dinte de ferestrău" pe intervalul 0-6π (Figura 1.6):» t=0:0.1:6\*pi;» x=sawtooth(t);» plot(t,x)

Procedura pulstran permite de a forma oscilaţii, care sunt secvenţe a impulsurilor dreptunghiulare, triunghiulare sau gaussiene. Adresarea către ea are forma:

>> t=0:0.1:6\*pi;

>> x=sawtooth(t);

>> plot(t,x);

>>grid;

Alta metoda de realizare:

>> t = 0:0.1:6\*pi;

>> x = 2 \* (t/(2\*pi) - floor(t/(2\*pi) + 0.5));

>> plot(t, x);

>> grid;

Procedura pulstran permite de a forma oscilaţii, care sunt secvenţe a impulsurilor dreptunghiulare, triunghiulare sau gaussiene. Adresarea către ea are forma:

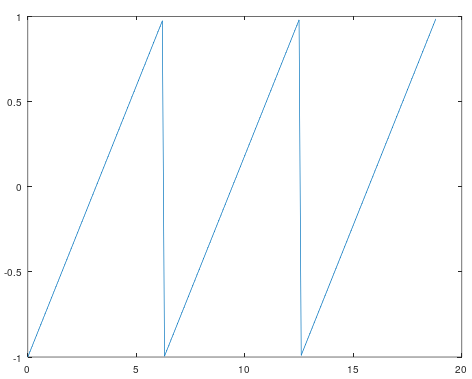


Figura 1.6 Crearea semnalelor "dinte de ferestrău ".

Aici d determină valoarea vectorului acelor momente de timp unde trebuie să fie centrele impulsurilor corespunzătoare; parametrul func determină forma impulsului şi poate avea una din următoarele sensuri: rectpuls (pentru impuls dreptunghiular), tripuls(pentru impuls triunghiular). gauspuls(sinusoidă modulată cu funcţia gauss). Semnalul de ieşire y se calculează pentru valorile argumentului, date în vectorul t, după formula:

y=func(t-d(1))+func(t-d(2))+...

Numărul impulsurilor în diapazonul dat a argumentelor este length(d). Parametrii p1, p2... determină parametrii necesari a impulsului în dependenţă de forma de adresare către procedura, care determină acest impuls.

Funcţia y=pulstran(t,d,p,Fs) permite de a determina impulsul cu secvenţa eşantioanelor date în vectorul p. Frecvenţa de discretizare este dată de parametrul Fs. La folosirea funcţiei y=pulstran(t,d,p), frecvenţa de discretizare se ia egală cu 1Hz. Mai jos sunt prezentate trei exemple de utilizare a procedurii pulstran:

Pentru o consecutivitate de impulsuri dreptunghiulare:

>> t=0:0.01:50;

>> d=[0:10:50];

>> y=0.6\*pulstran(t,d,'rectpuls',3);

>> plot(t,y),grid

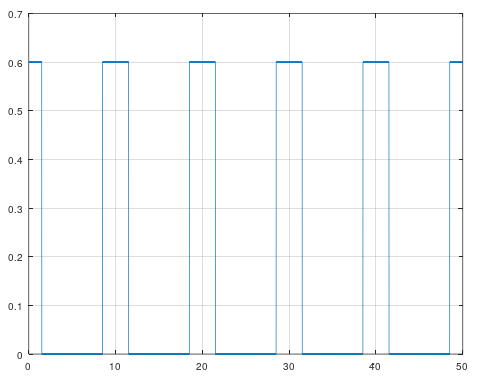


Figura 1.7 Utilizarea funcţiei pulstran la semnale dreptunghiulare

Pentru o consecutivitate de impulsuri triunghiulare:

>>t=0:0.01:50;

>>d=[0:10:50];

>>yl=0.8\*pulstran(t,d,'tripuls',5);

>>plot(t,yl);

>>grid;

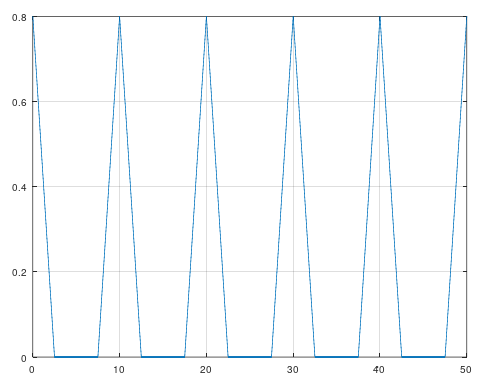


Figura 1.8 Utilizarea funcţiei pulstran la semnale triunghiulare.

Pentru o consecutivitate de impulsuri gaussiene:

>> t=0:0.01:50;

>> d=[0:10:50];

>> y2=0.7\*pulstran(t,d,'gauspuls',l,0.5);

>> plot(t, y2);

>> grid;

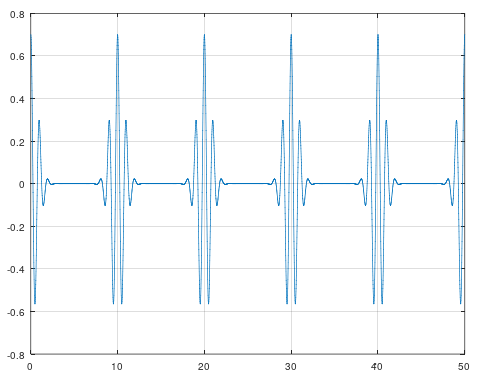


Figura 1.9 Utilizarea funcţiei pulstran la semnale gaussiene.

Formarea cosinusoidei frecvenţa căreia se schimbă liniar în timp, se face cu ajutorul procedurii chirp:

y=chirp(f,f0,t1,f1)

Ea formează eşantioanele din semnalul cosinusoidal cu frecvenţa schimbată liniar pentru momentele de timp date în vectorul t; f0-frecvenţa instantanee în momentul de timp t=0; f1-frecvenţa instantanee în momentul de timp t=l. Frecvenţele f0 şi f1 se dau în Hz. De exemplu f0=0, t1=l, f1 = 100.

>> t=0:0.001:l;

>> y=0.8\*chirp(t);

>> plot(t, y);

>> grid;

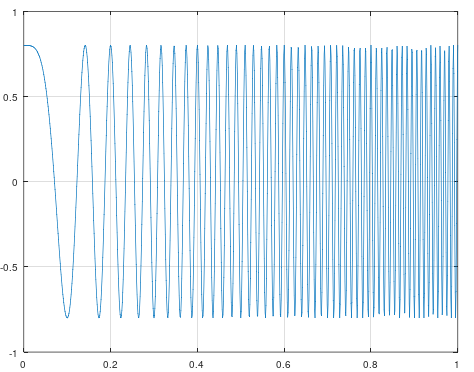


Figura 1.10 Formarea unei cosinusoide cu frecvenţa variabilă

1.8. Lucrul în laborator

Exemplul 1. Să se creeze din 256 eşantioane o oscilaţie armonică cu amplitudinea unitate şi perioada de 50 eşantioane. Pentru aceasta în regiunea de comandă MATLAB trebuie de cules următoarea serie de comenzi:

>> k=0:255;

>> x=sin(2\*pi\*k/50);

>> plot(x);

>> grid on;

>> title('sinusoida');

>> xlabel('numarul esantion');

>> ylabel('amplitudine')

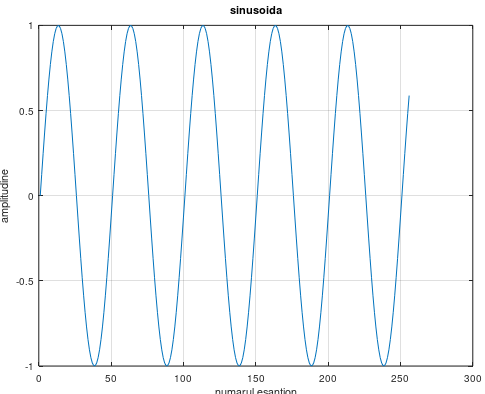


Figura 2.1

Exemplul 2. Să se creeze un semnal ce conţine 1024 eşantioane a unui proces tranzitoriu a unui sistem oarecare, descris de următoarea relaţie: Intervalul de timp între eşantioanele vecine T=0,001s. Formarea unui aşa tip de semnal este posibilă cu ajutorul următoarei serii de comenzi:

>> t=0:0.001:1.023;

>> T=0.001;

>> k=101:1024;

>> Z=zeros(1,100);

>> x=[Z exp(-(k\*T-0.1)/0.2).\*sin(2\*pi\*(k\*T-0.1)/0.16)];

>> plot(t,x);

>> grid on;

>> title('Proces tranzitoriu');

>> xlabel('Time(s)');

>> ylabel('Amplitudine');

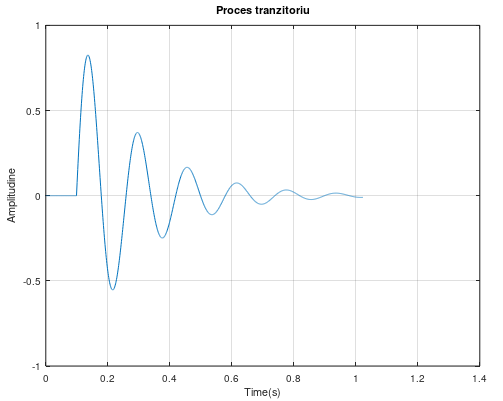


Figura 2.2

Exemplul 3. Să se creeze un semnal în intervalul de timp 0≤t≤1s care constă din suma unei oscilaţii armonice cu amplitudinea de 1V şi frecvenţa 50Hz, unei oscilaţii sinusoidale cu amplitudinea de 2V şi frecvenţa de 120Hz şi a unui semnal de zgomot distribuit normal cu valoarea medie zero şi valoarea medie pătratică 0.5V, folosind o frecvenţă de discretizare de 1000Hz, adică intervalul de discretizare T=0.001s.Setul de comenzi necesar pentru crearea unui astfel de semnal are forma:

>> t=0:0.001:1;

>> y=sin(2\*pi\*50\*t)+2\*sin(2\*pi\*120\*t);

>> randn('state',0);

>> yn=y+0.5\*randn(size(t));

>> plot(t(1:50), yn(1:50));

>> grid

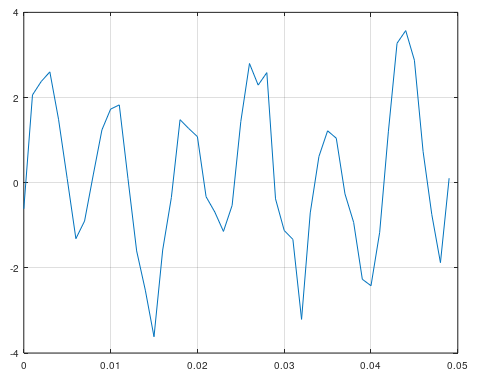


Figura 2.3

Exemplul 4. Să se creeze în intervalul 0≤t≤ls semnale exponenţiale de forma: a) 5exp(-6t); b) exp(5t), folosind frecvenţa eşantioanelor fd=1000Hz. Succesiunea comenzilor poate fi următoarea:

**a)>>** B=5;

>> a=6;

>> t=0:0.001:1;

>> x=B\*exp(-a\*t);

>> plot(t,x);

>> grid

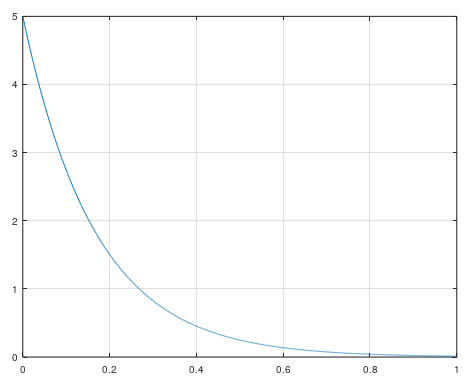


Figura 2.4

**b)** >> t=0:0.001:1;

>> x=B\*exp(-a\*t);

>> plot(t,x), grid

>> B=1;

>> a=5;

>> t=0:0.001:1;

>> x=B\*exp(a\*t);

>> plot(t,x);

>> grid

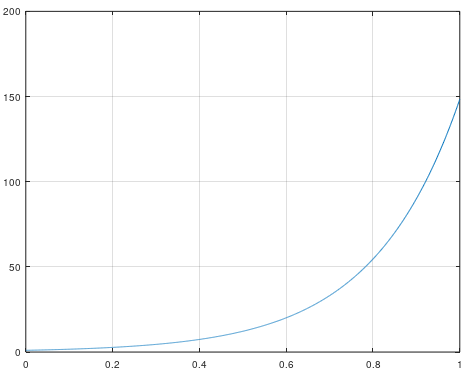


Figura 2.5

Exemplul 5. Să se creeze un impuls exponenţial discret de forma x(n)=Brn, unde B=l, r=0.8 pe intervalul -10≤n≤10. Pentru aceasta pot fi folosite următoarele comenzi:

>> B=1;

>> r=0.8;

>> n=-10:10;

>> x=B\*r.^n;

>> stem(n,x)

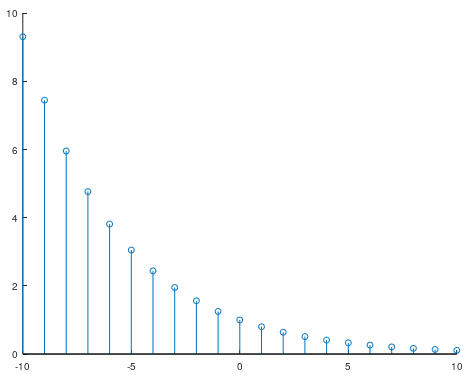
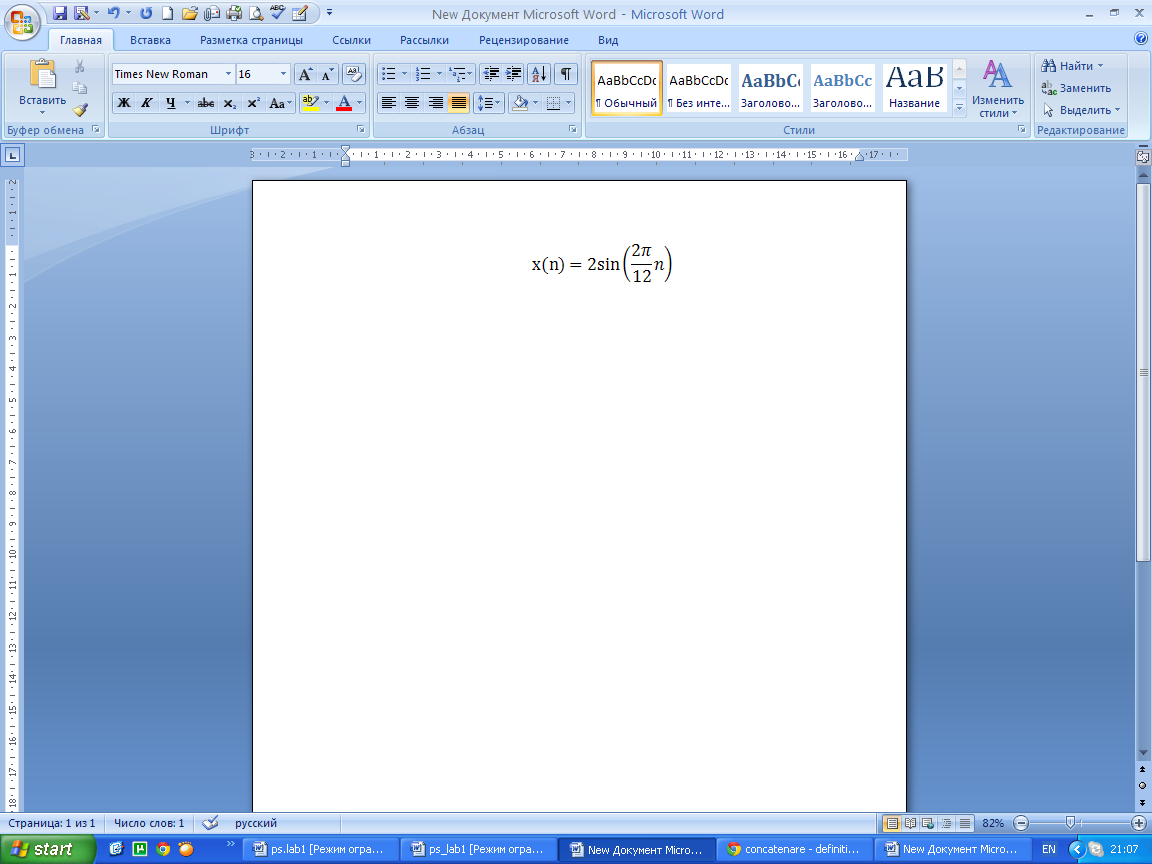


Figura 2.7

Exemplul 6. Să se creeze un semnal sinusoidal discret cu perioada de 12 eşantioane şi amplitudinea de 2V:

10≤n≤10

Pentru aceasta poate fi folosit următorul set de comenzi:

>> A=1;

>> omega=2\*pi/12;

>> n=-10:10;

>> y=A\*sin(omega\*n);

>> stem(n,y)

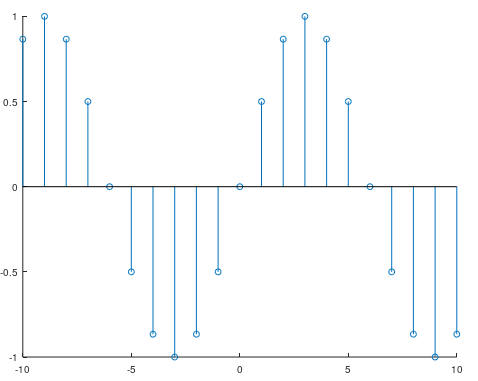


Figura 2.8

Exemplul 7. Să se creeze un semnal discret exponenţial atenuat, pe baza înmulţirii exponentei atenuatoare formate în exemplul 5 şi a semnalului sinusoidal creat în exemplul 6, ambele primitei pentru intervalul -10≤n ≤10. Soluţie: notând rezultatul înmulţirii cu Z(n) putem folosi următoarele comenzi pentru generarea şi vizualizarea semnalului:

>> Z=x.\*y;

>> stem(n,Z)

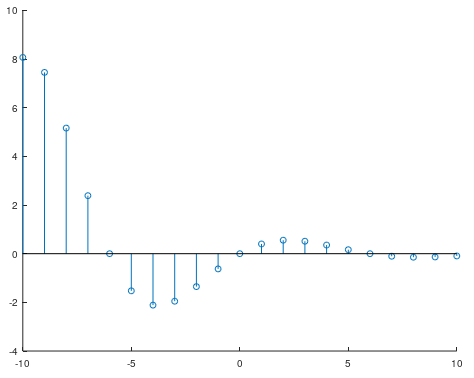
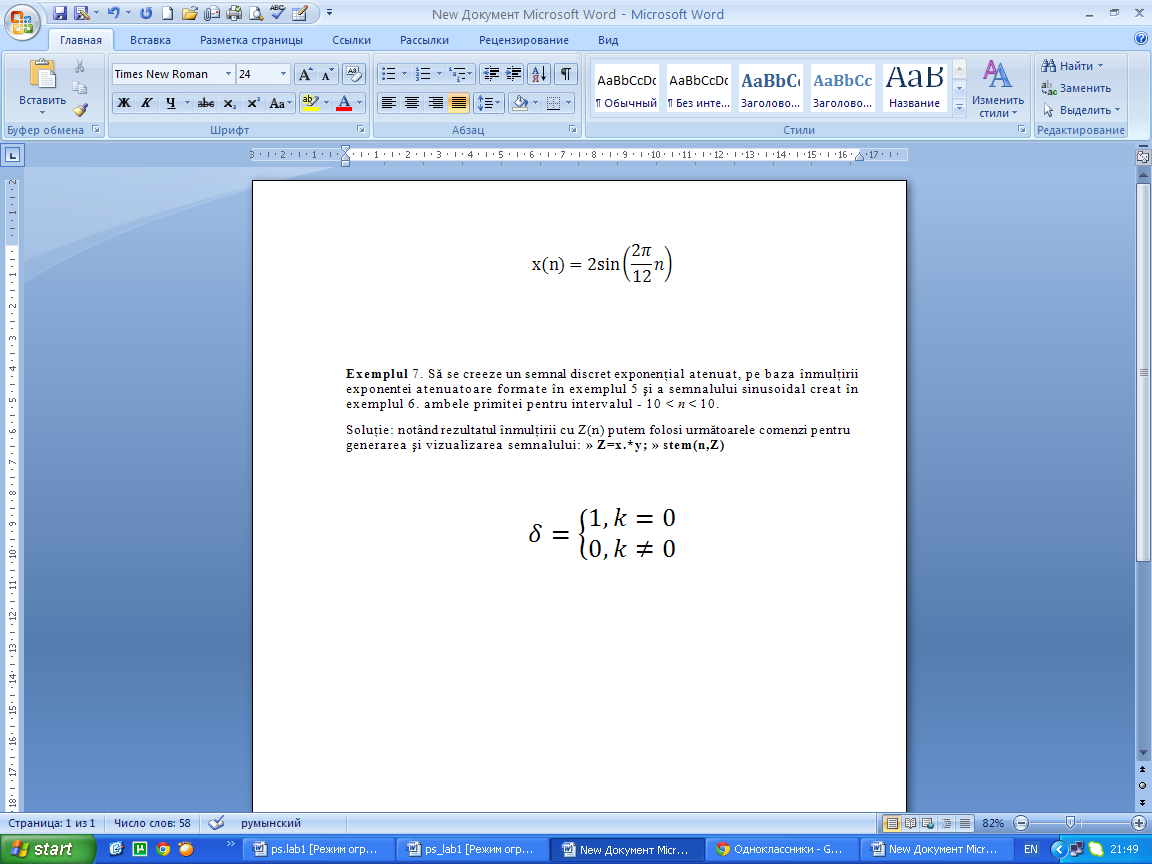


Figura 2.9

Exemplul 8. Să se creeze următoarele serii de impulsuri δ, care se descriu, după cum se ştie, de relaţia:



a) x1=2δ(k-3), 1≤k≤10;

b) x2=0.6δ(k), -10≤k≤10 ;

c) x3=1.5δ(k+7), -10≤k≤0

a) >> k=1:10;

>> x1=zeros(size(k));

>> x1(3)=2;

>> figure;

>>stem(k,x1);

>>grid;

>>xlabel('k');

>>ylabel('x1(k)');

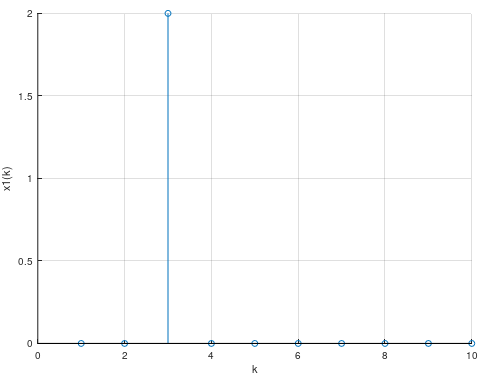


Figura 2.10

b) >> k=-10:10;

>> x2=zeros(size(k));

>> x2(11)=0.6;

>> figure;

>> stem(k,x2);

>> grid;

>> xlabel('k');

>> ylabel('x2(k)');

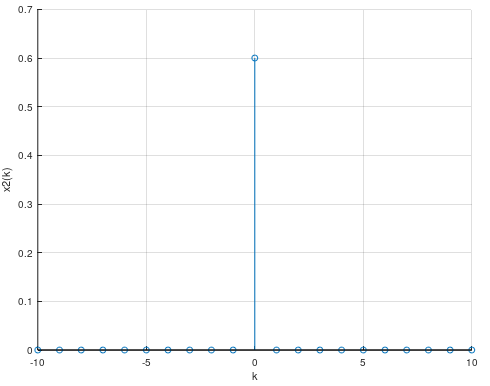


Figura 2.11

c) >> k=-10:0;

>> x3=zeros(size(k));

>> x3(4)=1.5;

>> figure;

>> stem(k,x3);

>> grid;

>> xlabel('k');

>> ylabel('x3(k)');

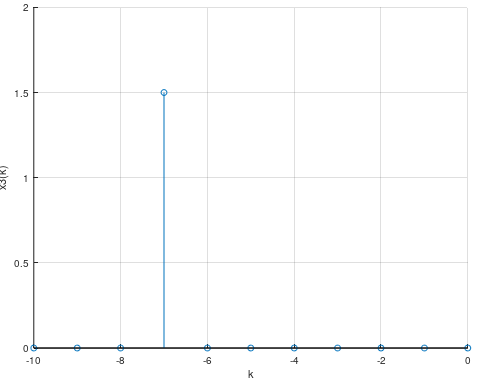


Figura 2.12

Exemplul 9. Crearea impulsului dreptunghiular de amplitudine unitate şi durata 1s, amplasat simetric în originea de coordonate t=0 (-0.5s≤t≤0.5s) descris în intervalul dc timp -1s≤t≤1s cu pasul T=2ms.Soluţie: impulsul dreptunghiular poate fi creat cu ajutorul diferenţei a două funcţii de tip "treaptă" deplasate în timp cu un interval egal cu durata impulsului. Acest semnal poate fi format cu ajutorul următorului set de comenzi:

>> t=-1:0.002:1;

>> U1=[zeros(1,250), ones(1,751)];

>> U2=[zeros(1,751), ones(1,250)];

>> U=U1-U2;

>> plot(t,U);

>> grid;

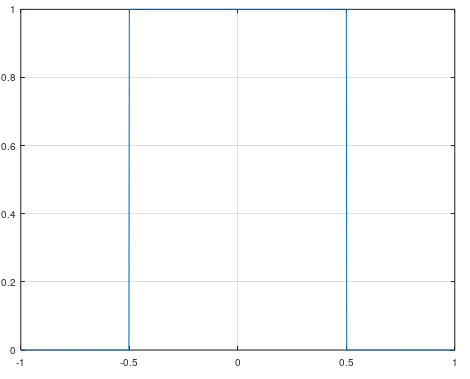


Figura 2.13

Exemplul 10. Crearea unui şir discret de impulsuri dreptunghiulare cu amplitudinea A=l şi viteza unghiulară ω=π/4 în intervalul. -10≤n≤10 se efectuează cu ajutorul următoarelor comenzi:

>> A=1;

>> omega=pi/4;

>> rho=0.5; %parametrul rho determina partea perioadei in care semnalul e pozitiv

>> n= -10:10;

>> x=A\*square(omega\*n\*rho);

>> x = A \* sign(sin(omega \* n \* rho)); % sau varianta data.

>> stem(n,x)

>> grid

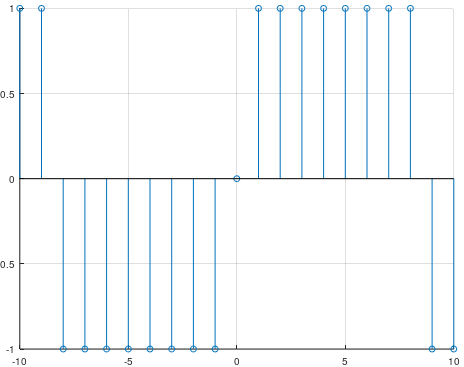


Figura 2.14

Exemplul 11. Crearea semnalului exponenţial complex

Poate fi folosit următorul set de comenzi:



>> n=0:25;

>> x=exp(j\*n/3);

>> subplot(2,1,1);

>> stem(n,real(x));

>> title('real part');

>>xlabel('index(n)');

>> subplot(2,1,2);

>> stem(n,imag(x));

>> title('imag part');

>>xlabel('index(n)');

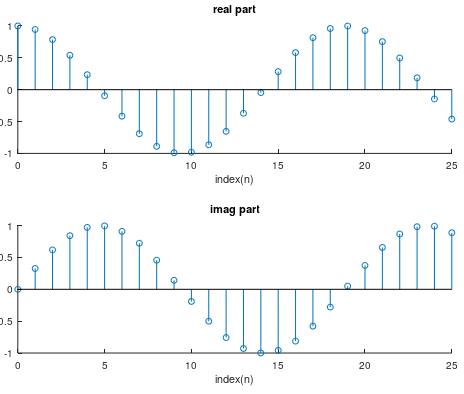


Figura 2.15

Exemplul 12. Crearea semnalului exponenţial complexx(n)=exp((-0.1+j0.3)n) pe intervalul -10≤n≤10:

>> n=[-10:1:10];

>> alpha=-0.1+0.3j;

>> x=exp(alpha\*n);

>> subplot(2,2,1);

>> stem(n,real(x));

>> title('real part');

>> xlabel('n');

>> grid;

>> subplot(2,2,2);

>> stem(n,imag(x));

>> title('imaginare part');

>> grid;

>> xlabel('n');

>> subplot(2,2,3);

>> stem(n,abs(x));

>> title('magnitudine');

>> xlabel('n');

>> grid;

>> subplot(2,2,4);

>> stem(n,(180/pi)\*angle(x));

>> title('phase');

>> xlabel('n');

>> grid;

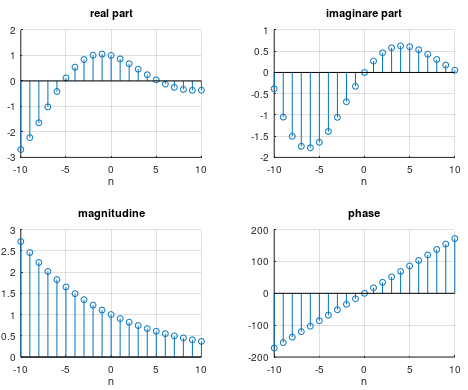


Figura 2.16

**Concluzie:**

În prelucrarea semnalelor digitale, este adesea necesar să separăm un semnal de zgomot sau să îmbunătățim calitatea semnalului în prezența zgomotului, ceea ce reprezintă o provocare în domenii precum telecomunicațiile și prelucrarea semnalelor audio. Pentru a atinge acest obiectiv, se folosesc diverse tehnici și algoritmi.Un exemplu specific de prelucrare a semnalelor implică un semnal distorsionat de zgomot aleatoriu, iar obiectivul este să obținem un semnal cât mai aproape de original eliminând zgomotul. Pentru aceasta, se utilizează un algoritm simplu de filtrare, în care fiecare punct al semnalului rezultat este calculat ca medie a sa și a două puncte vecine.Semnalele sinusoidale sunt esențiale în prelucrarea semnalelor și pot fi generate în MATLAB folosind funcțiile sin și cos. Aceste semnale sunt utile în telecomunicații, audio și alte domenii pentru analiză, sinteză și manipulare. De asemenea, secvențele exponențiale sunt semnale digitale importante, generate în MATLAB cu operatorii .^ și exp. Acestea joacă un rol semnificativ în analiza semnalelor, în special în transformatele Fourier și în domeniul filtrelor.