Universitatea Tehnica din Republica Moldova

Facultatea Calculatoare, Informatica si Microelectronica

Departament Inginerie Software si Automatica

Specialitatea Tehnologia Informației

Raport

Curs: Prelucrarea semnalelor

Tema: Introducere în Matlab

A elaborat: Reguș Ruslan Grupa: TI-214

A verificat: Asist. Univ. Cazac A.

Chișinău 2023

**Lucrare de laborator nr. 1**

**Obiective:** însușirea elementelor de bază din sistemul MATLAB precum: definirea unei matrice, operații cu matrice, tipuri de date, variabile, instrucțiuni, fișiere script, reprezentarea graficelor și implementarea funcțiilor în Matlab.

**Exercitiul 1.1**

Exerciţiu: Fie vectorii linie a = [0, 0.1, 0.2, … , 2] şi coloană b = [ 1 ⋮ 1 ]

a) Ce lungime trebuie să aibă b astfel, încât să aibă sens înmulţirea (în sens matricial) a\*b? Iniţializaţi în MATLAB cei doi vectori şi efectuați înmulţirea.

Se creează un vector numit a folosind funcția linspace, care împarte intervalul [0,2] în 21 de puncte egale și stochează aceste valori în vectorul a.Apoi, se creează un alt vector numit b, care este un vector coloană de 21 de elemente, fiecare element fiind 1.În cele din urmă, se efectuează o operație de înmulțire între matricele a și b, utilizând operatorul \*, care returnează un scalar. Această operație este posibilă deoarece b este un vector coloană și a este un vector linie, astfel încât se poate face înmulțirea dintre o matrice linie și o matrice coloană. Rezultatul este un scalar, egal cu suma elementelor vectorului a. v=linspace(minim,maxim,număr\_de\_elemente) – se generează un vector linie v cu elementele, începând de la minim la maxim, cu pas constant şi având un număr de elemente egal cu număr\_de\_elemente.

|  |
| --- |
| a = linspace(0, 2, 21);  b = ones (21, 1);  a\*b |

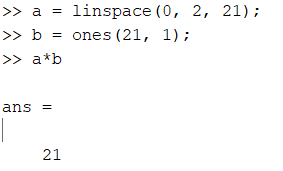


Fig 1.1 Înmulțirea a\*b

b) Efectuați înmulţirea b\*a. Execuția în Fig 1.2.

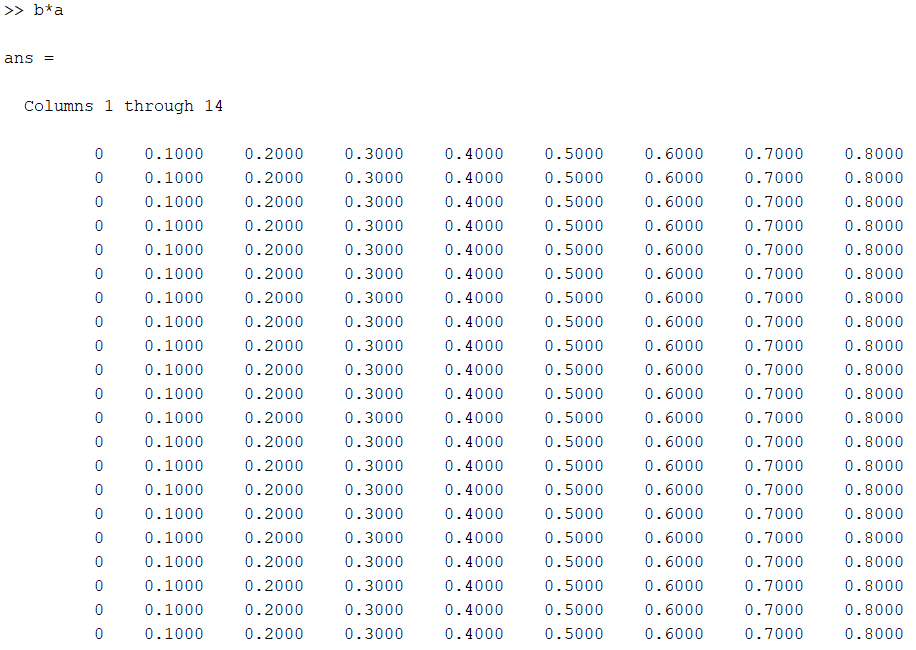


Fig 1.1 Înmulțirea b\*a

În Fig 1.2 se observă că înmulțirea b\*a crează o matrice de 21x21 în fiecare rând fiind fiecare număr din matricea a.

**Exercitiul 1.2**

Exerciţiu: Se creează un fişier nou care trebuie salvat în directorul d:/student/pns/nrgrupa. Folosind sintaxele şi indicaţiile din secţiunile 1.2.6 şi 1.2.7, elaborați un program MATLAB care să genereze un vector cu elemente aleatoare cu distribuţie normală (gaussiană) şi să afişeze elementele negative ale acestui vector.

Randn(m,n) – returnează o matrice de dimensiune m x n, având drept elemente numere aleatoare cu distribuţie normală de medie nulă şi varianţă unitară;

|  |
| --- |
| n = 50;  v = randn(n, 1);  for i = 1:n  if v(i) < 0  disp(v(i));  end  end |

În codul de mai sus este mai întăi o variabilă ce conține numărul de date din vector n = 50. După care se generează vectorul v cu elemente aleatoare cu distribuție normală. E utilizează sintaxa for pentru a parcurge întreg vectorul, iar apoi se verifică dacă e negativ se afișează, daca nu, trece la următorul.

Acest cod generează un vector aleatoriu v cu n elemente, distribuite normal (adica cu media zero si deviatia standard de 1), utilizand functia randn din MATLAB. Apoi, utilizeaza o structura de bucla for pentru a parcurge fiecare element al vectorului si afiseaza elementele negative ale vectorului. Mai exact, in fiecare iteratie a buclei, se verifica daca valoarea elementului curent al vectorului este mai mica decat zero si, in caz afirmativ, se afiseaza aceasta valoare folosind functia disp.



Fig 1.3 Execuția codului exercițiului 2

În Fig 1.3 se poate observa că la executarea scriptului s-au obținut doar numere negative.

**Exercitiul 1.3**

Exerciţiu: Elaborați un program MATLAB care generază un vector cu elemente complexe.

Elaborați (un alt fişier) o funcţie MATLAB care, având drept parametru de intrare vectorul cu valori complexe, returnează ca parametri de ieşire:

- media aritmetică a părţilor reale ale elementelor vectorului;

- un vector ce conţine elementele vectorului iniţial ridicate la pătrat;

- o matrice obţinută din înmulţirea vectorului iniţial cu transpusul său.

Atenţie: pentru a nu se afişa rezultate intermediare din funcţie sau elementele unor variabile se va folosi ; la sfârşitul liniei respective de program.

rand(m,n) – returnează o matrice de dimensiune m x n, având drept elemente numere aleatoare cu distribuţie uniformă între 0 şi 1;

Cod generare vector cu numere complexe, primul program:

|  |
| --- |
| function v = function1()  n = 5;  re = rand(n, 1);  im = rand(n, 1);  v = complex(re, im);  disp("Vectorul v");  disp(v);  end |

Acest cod definește o funcție numită "function1" care generează un vector aleatoriu de numere complexe.În primul rând, se definește o variabilă "n" cu valoarea 5. Apoi, se generează două vectori aleatorii, "re" și "im", fiecare cu n elemente, folosind funcția "rand". Vectorul "re" conține numere aleatoare între 0 și 1, iar vectorul "im" conține numere aleatoare între -1 și 1.Apoi, se construiește un vector de numere complexe, "v", folosind funcția "complex", care primește ca argumente vectorii "re" și "im".În cele din urmă, se afișează vectorul "v" folosind funcția "disp". Când această funcție este apelată din afara funcției "function1", vectorul "v" este returnat ca rezultat al funcției.

Cod executare cerințe, al doilea program:

|  |
| --- |
| run('Lab1GenerareVComplexe.m');  [med, patr, mat] = function2(v);    disp("Media aritmetica");  disp(med);  disp("Ridicare la patrat");  disp(patr);  disp("Inmultire cu transpusul");  disp(mat);      function [med, patr, mat] = function2(vect)  med = mean(real(vect));  patr = vect.^2;  mat = vect \* vect';  end |

În primul cod se generează un vector cu numere complexe care se salvează într-o variabilă v, iar al doilea efectuează operațiile cerute în condiție, unde variabila v este transmisă de la primul cod la apelarea programului Lab1GenerareVComplexe.m. Mean face media aritmetica, iar real extrage partea reala. Execuția codului se poate observa în Fig 1.4.

Acest cod conține două funcții și un apel de funcție.Prima funcție, function1(), generează un vector complex v cu 5 elemente aleatoare, afișează vectorul și îl returnează.Apoi, în apelul de funcție, se apelează funcția function2() cu vectorul generat v. Funcția function2() primește vectorul ca argument și returnează media aritmetică a părții reale a vectorului, un vector format din pătratele fiecărui element al vectorului dat și o matrice obținută prin înmulțirea vectorului cu transpusul său.În cele din urmă, valorile returnate de function2() sunt asignate la trei variabile med, patr și mat, care sunt afișate la ieșirea din program.

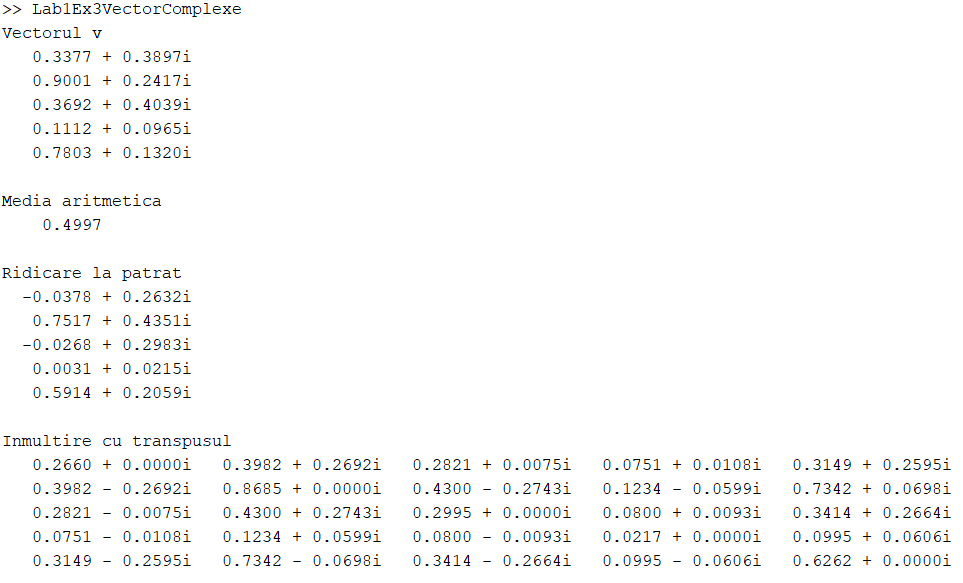


Fig 1.4 Execuția ex3 MATLAB

**Exercitiul 1.4**

Exerciţiu Elaborați un program MATLAB în care să generaţi şi să reprezentaţi grafic folosind funcţia stem următorii vectori:

a) z = [0,0,0,0,0,1,0,0,...,0], vectorul z, având lungimea 21. Reprezentarea grafică se va face în două „miniferestre” (funcția subplot), vectorul z în funcție de n=0:20, respectiv de m=-5:15;

b) t=|10-n |, reprezentat graphic în fincție de n=0:20;

c) 𝑥1 = 𝑠𝑖𝑛 ( 𝜋 17 𝑛), -15≤n≤25 și 𝑥2 = 𝑐𝑜𝑠 ( 𝜋 √23 𝑛), 0≤n≤50.

Cele două secvenţe vor fi reprezentate:

- în figura 1 – în acelaşi sistem de coordonate (pe acelaşi grafic);

- în figura 2 – folosind două “miniferestre” grafice plasate una sub alta. Reprezentaţi cele două figure, folosind comanda plot, apoi încercaţi reprezentarea lor, folosind comanda stem.

Cu funcţia plot se pot reprezenta grafic semnale sau funcţii “continue”, deoarece se unesc cu linie continuă valorile care se reprezintă. Astfel, se pot reprezenta semnale continue, alegând variabila timp cu pasul mai mic decât variaţia semnalului reprezentat. De exemplu, dacă perioada semnalului e 0.01 secunde se poate alege variabila temporală cu pasul de 0.001s: t = 0:0.001:5 (secunde).

Exemplu: Să se reprezinte grafic cu funcţia plot un semnal sinusoidal de frecvenţă 50 Hz, de durată 0.2 secunde şi amplitudine 2. Se va alege rezoluţia temporală 1ms. 32 F = 50; t = 0:0.001:0.2; s = 2\*sin(2\*pi\*F\*t); plot(t,s,'.-'),xlabel('Timp [s]'),grid

0:20 = i-a toate numere de la 0 la 20

Cod exercițiu 4

|  |
| --- |
| n=0:20;  m=-5:15;  z = zeros(21);  z(6) = 1;    subplot(2, 1, 1)  stem(n, z)  subplot(2, 1, 2)  stem(m, z)    t=abs(10-n);  figure(2)  stem(n,t)    n1=-15:25;  x1=sin(pi/17\*n1);  n2=0:50;  x2=cos(pi/sqrt(23)\*n2);  figure(3)  plot(n1,x1,'g',n2,x2,'r')    figure(4)  subplot(2,1,1)  plot(n1,x1)  subplot(2,1,2)  plot(n2,x2)    figure(5)  stem(n1,x1)  hold on  stem(n2,x2)  hold off    figure(6)  subplot(2,1,1)  stem(n1,x1)  subplot(2,1,2)  stem(n2,x2) |

Acest cod MATLAB conține o serie de comenzi pentru a genera și afișa diferite grafice.

În prima parte a codului, se definește două vectori n și m și se creează un vector nul z cu dimensiunea 21. Un singur element al vectorului z este setat la 1 și acesta este afișat folosind stem() în două subplot-uri diferite.

În continuare, se calculează un alt vector t prin scăderea valorii absolute a diferenței dintre 10 și fiecare element din vectorul n. Acesta este, de asemenea, afișat folosind stem().

Următoarele două secțiuni de cod sunt folosite pentru a genera și afișa grafice de funcții sinus și cosinus. În primul rând, se definește n1 ca fiind intervalul de la -15 la 25, iar x1 este definit ca o funcție sinus cu o perioadă de 17. Apoi, se definește n2 ca intervalul de la 0 la 50, iar x2 este definit ca o funcție cosinus cu o perioadă de sqrt(23). Funcțiile x1 și x2 sunt apoi afișate folosind plot() într-un grafic cu două culori diferite.

În următoarele două secțiuni de cod, graficele sunt afișate separat pentru fiecare funcție.

În ultima secțiune de cod, cele două grafice separate sunt afișate folosind stem() în două subplot-uri separate.

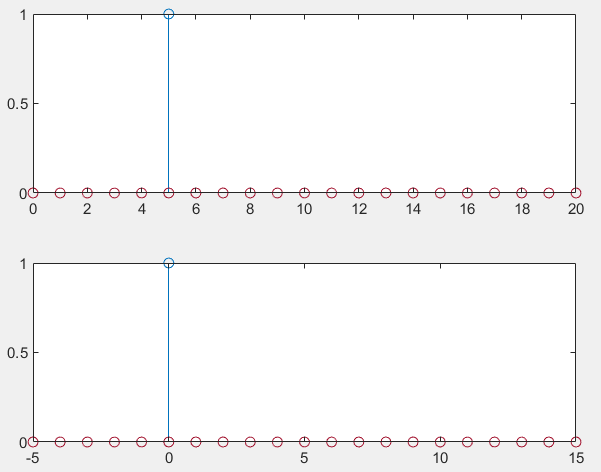


Fig 1.5 Partea a) subplot pentru n și m pentru vectorul z

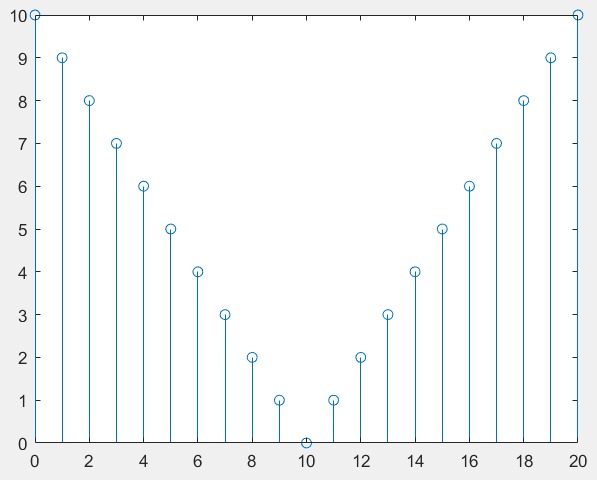


Fig 1.6 Partea b) reprezentarea grafică în funcție de n=0:20.

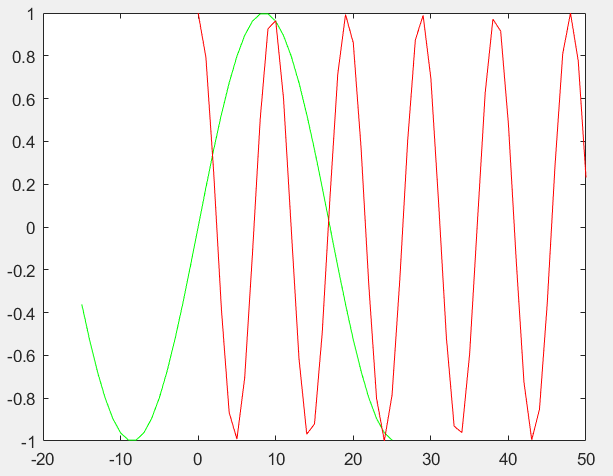


Fig 1.7 Partea c) reprezentarea grafică în acelaşi sistem de coordonate cu subplot.

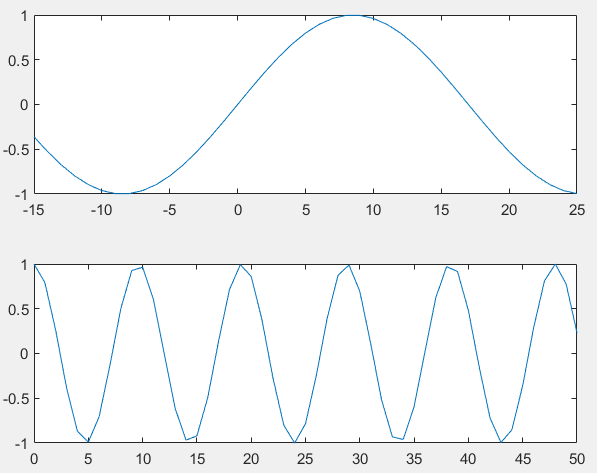


Fig 1.8 Partea c) reprezentarea grafică folosind două “miniferestre” cu subplot.

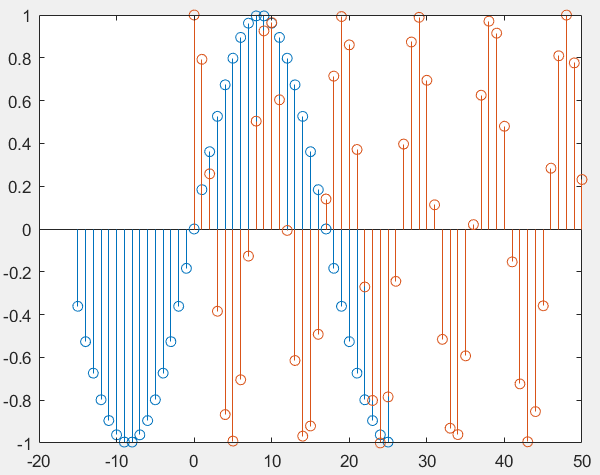


Fig 1.9 Partea c) reprezentarea grafică în acelaşi sistem de coordonate cu stem.

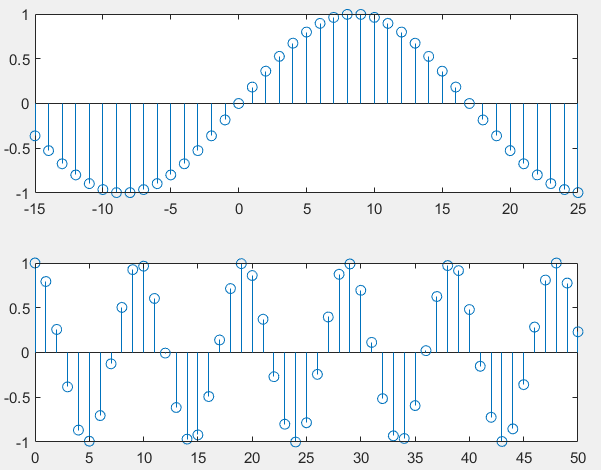


Fig 1.10 Partea c) reprezentarea grafică folosind două “miniferestre” cu stem.

**Exercitiul 1.5**

Exerciţiu:

1) Modificaţi pasul de variaţie a variabilei t la 0.01, apoi la 0.0002. Comentaţi diferenţele;

2) Măsuraţi pe grafic perioada semnalului sinusoidal în cele 3 situaţii;

3) Generaţi un semnal cosinusoidal de frecvenţă 20 Hz pe care să-l reprezentaţi cu culoare roşie pe acelaşi grafic peste semnalul sinusoidal.

Cod Exerciţiu 5

|  |
| --- |
| F = 50;  t = 0:0.001:0.2;  s = 2\*sin(2\*pi\*F\*t);    figure (1) ;  subplot (1,3,1);  plot(t,s,'.-'),xlabel('Timp [s]'),grid    t = 0:0.01:0.2;  s = 2\*sin(2\*pi\*F\*t);  subplot (1,3,2);  plot(t,s,'.-'),xlabel('Timp [s]'),grid    t = 0:0.0002:0.2;  s = 2\*sin(2\*pi\*F\*t);  subplot (1, 3, 3);  plot(t,s,'-'),xlabel('Timp [s]'),grid |

Acest cod generează și afișează trei semnale sinusoidale diferite în funcție de rezoluția timpului.În primul rând, se stabilește frecvența de 50 Hz și se creează un vector de timp t între 0 și 0,2 secunde cu o rezoluție de 0,001 secunde. Apoi, se generează un semnal sinusoidal s cu o amplitudine de 2, folosind funcția sin() și se afișează într-un grafic cu subplot (1, 3, 1).Apoi, se generează același semnal sinusoidal s, dar cu o rezoluție de timp mai mică, de 0,01 secunde, și se afișează într-un alt subplot (1, 3, 2).În cele din urmă, se generează din nou același semnal sinusoidal s, dar cu o rezoluție de timp și mai mică, de 0,0002 secunde, și se afișează într-un al treilea subplot (1, 3, 3).În toate cele trei subplot-uri, se afișează graficul semnalului s în funcție de timpul t, cu etichetarea axei x cu "Timp [s]" și cu o grilă de fundal pentru a face graficul mai ușor de citit.

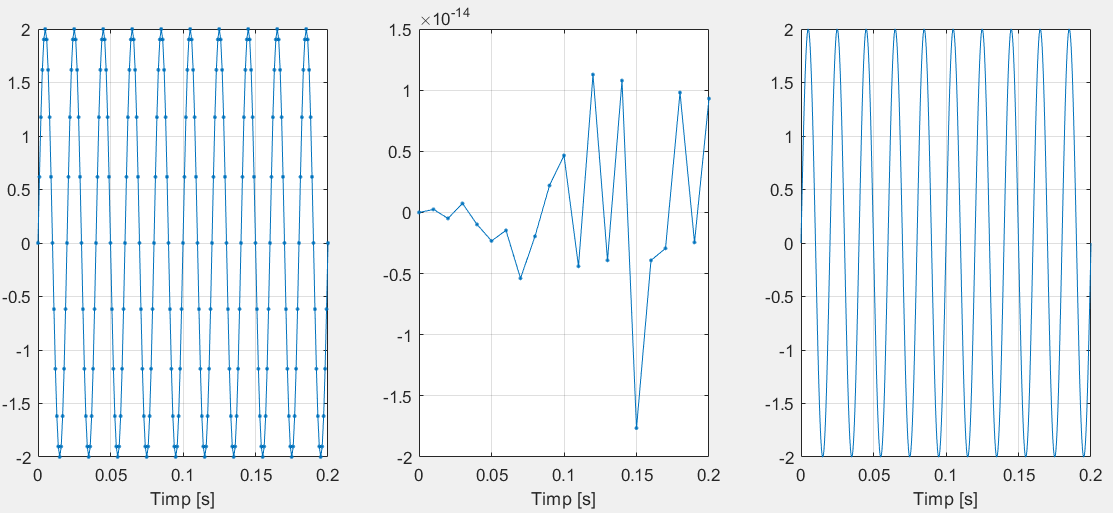


Fig 1.11 Execuția codului ex 5

|  |
| --- |
| F=50;  F2=20;  t=0:0.0002:0.2;  s=2\*sin(2\*pi\*F\*t);  c=2\*cos(2\*pi\*F2\*t);  plot(t,s,'.-'),xlabel('Timp [s]'),grid  hold on  plot(t,c,'.-'),xlabel('Timp [c]'),grid |

Acest cod generează două semnale sinusoidale, unul cu frecvența F = 50 Hz și celălalt cu frecvența F2 = 20 Hz, și le afișează pe aceeași grafică.Variabila t reprezintă vectorul de timp, cu o rezoluție de 0.0002 secunde, de la 0 la 0.2 secunde.Semnalul sinusoidal cu frecvența F și amplitudinea 2 este generat cu ajutorul funcției sin() și este stocat în variabila s.Semnalul sinusoidal cu frecvența F2 și amplitudinea 2 este generat cu ajutorul funcției cos() și este stocat în variabila c.Apelul funcției plot() este folosit pentru a afișa cele două semnale pe aceeași figură. Funcția xlabel() este folosită pentru a afișa eticheta axei x. Funcția grid() este folosită pentru a afișa grila pe grafic.Comanda hold on este folosită pentru a păstra graficul anterior pe ecran și a permite adăugarea unui alt grafic pe aceeași figură, în acest caz semnalul sinusoidal cu frecvența F2. Comanda hold off ar fi necesară pentru a dezactiva această funcționalitate dacă dorim să adăugăm un nou grafic.

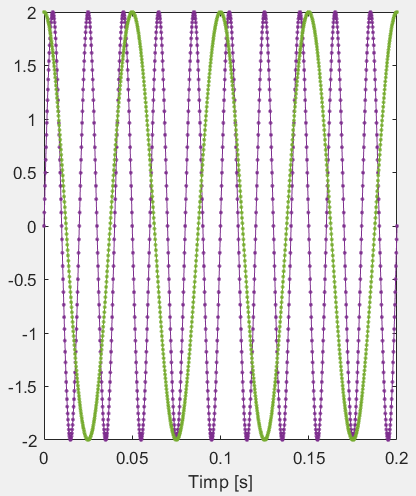


Fig 1.11 Execuția codului ex 5 punctul c)

**Exercitiul 1.6**

Exerciţiu: Să se genereze vectorul x conținând valorile 1, 2, 3, ..., 99, 100 și vectorul y având valorile 2, 4, 6, 8, ..., 198, 200:

a) să se reprezinte grafic y în funcție de x, folosind funcția stem;

b) să se reprezinte grafic y în funcție de x, folosind funcția plot;

c) să se reprezinte grafic în aceeași figură, în același sistem de coordonate, y în funcție de x, folosind funcțiile plot și stem (se vor folosi culori diferite);

d) să se reprezinte grafic în aceeași figură, în sisteme de coordonate diferite, y în funcție de x, folosind funcția plot și y în funcție de x, folosind funcția stem.

|  |
| --- |
| x=1:1:100;  y=2:2:200;  stem(y); |

Acest cod creează două vectori, x și y, care conțin valori consecutive de la 1 la 100, respectiv de la 2 la 200 cu un pas de 2. Apoi, se afișează o diagramă de tip stem a vectorului y utilizând funcția stem(). Diagrama de tip stem este o reprezentare grafică a datelor care afișează punctele de date deasupra axei orizontale, iar o linie verticală de la fiecare punct de date coboară la axa orizontală. Dacă un argument este dat pentru stem(), atunci acesta reprezintă valorile pentru axa orizontală și argumentul dat reprezintă valorile pentru axa verticală. Dacă nu se specifică niciun argument, atunci stem() va utiliza vectorul dat pentru a construi axa orizontală, începând cu 1 și adăugând un pas de 1 pentru fiecare element al vectorului.

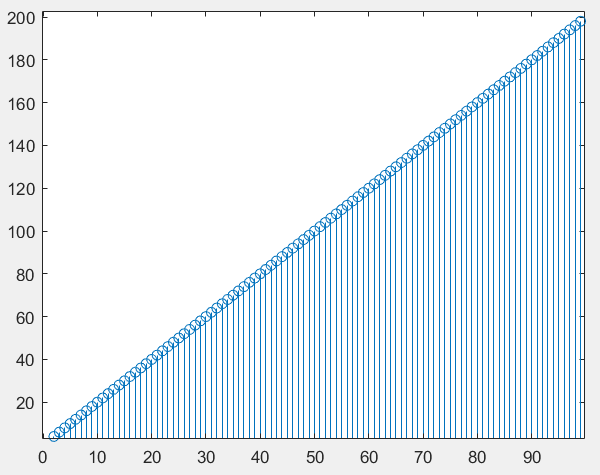


Fig 1.12 Reprezintare grafica y în funcție de x, folosind funcția stem

|  |
| --- |
| x=1:1:100;  y=2:2:200;  plot(y); |

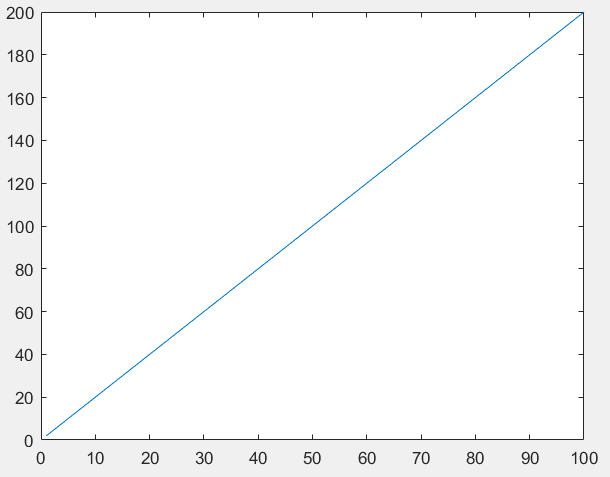


Fig 1.13 Reprezintare grafica y în funcție de x, folosind funcția plot

|  |
| --- |
| x=1:1:100;  y=2:2:200;  stem(x);  hold on  plot(y); |

În acest cod, variabila x este un vector care conține numerele întregi de la 1 la 100 cu o diferență de 1 între elemente, iar variabila y este un vector care conține numerele întregi de la 2 la 200 cu o diferență de 2 între elemente.În prima linie din blocul de cod, funcția stem() este apelată cu argumentul x, ceea ce înseamnă că va fi afișat un grafic cu o reprezentare punctată a datelor din x.Apoi, cu ajutorul comenzii hold on, se păstrează graficul anterior și se adaugă un alt grafic, reprezentând datele din y. Pentru aceasta, se folosește funcția plot() cu argumentul y, ceea ce înseamnă că graficul va fi reprezentat printr-o linie continuă.În final, cu comanda hold off, se dezactivează modul hold, astfel încât următorul grafic afișat să fie reprezentat fără a mai fi adăugat peste cele precedente.

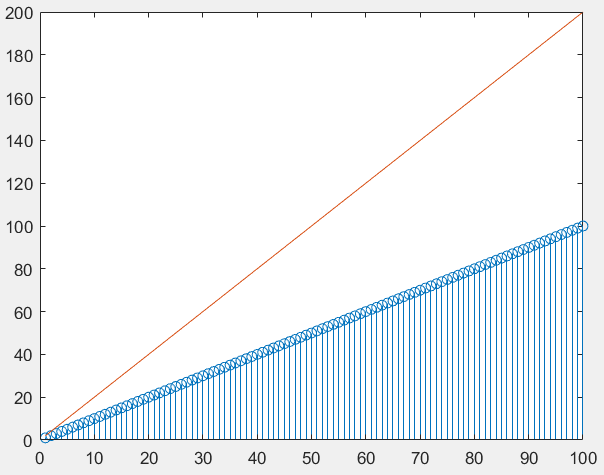


Fig 1.14 Reprezintare grafica y față de x, folosind funcția plot și stem

**Exercitiul 1.7**

Exerciţiu:

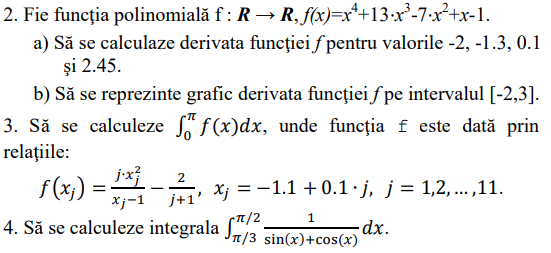
1. Fie f o funcţie reală de o variabilă reală, precizată de următorii vectori: x=[-2 -1.5 0 0.5 2.5 3]; y=[6 9 11 10 7 5]. Să se aproximeze derivata de ordinul I a funcţiei f în punctele:

a) -2, 0 şi 2.5, folosind diferenţe progresive;

b) -1.5, 0 şi 3, folosind diferenţe regresive;

c) -0.75, 0.25, 0.5 şi 1, folosind diferenţe centrate.

Să se reprezinte grafic în aceeaşi fereastră grafică derivata de ordinul I a funcţiei f, obţinută prin aproximare cu cele trei tipuri de diferenţe finite.



Codul exercitiului 7 punctul 1)

|  |
| --- |
| x=[-2 -1.5 0 0.5 2.5 3];  y=[6 9 11 10 7 5];  dx=diff(x);  dy=diff(y);  %a)  dif=dy./dx;  px=[-2 0 2.5];  disp('Diferenta progresiva');  for i=1:length(px)  temp = find(x == px(i));  disp([' f''(' num2str(px(i)) ')=' num2str(dif(temp))]);  end  %b)  rx=[-1.5 0 3];  disp('Diferenta regresiva');  for i=1:length(rx)  temp = find(x == rx(i));  disp([' f''(' num2str(rx(i)) ')=' num2str(dif(temp - 1))]);  end  %c)  lenx=length(dx);  difc=(dy(1:lenx-1)+dy(2:lenx))./(dx(1:lenx-1)+dx(2:lenx));  cx=[-0.75 0.25 0.5 1];  disp('Diferenta regresiva');  for i=1:length(cx)  temp = find(x == cx(i));  disp([' f''(' num2str(cx(i)) ')=' num2str(difc(temp))]);  end |

Acest cod calculează derivatele unei funcții definite prin vectorii x și y, prin utilizarea diferențelor finite progresive, regresive și centrale.În primul rând, se calculează diferențele finite ale vectorilor x și y prin intermediul funcției diff(). Apoi, se calculează diferențele finite progresive, regresive și centrale ale funcției utilizând formulele corespunzătoare.Pentru a afișa valorile derivatei în punctele de interes, se definesc vectorii px, rx și cx, care conțin aceste puncte și se caută valorile derivatei în aceste puncte prin căutarea poziției lor în vectorul x cu funcția find().Valorile derivatei sunt afișate la ieșirea din program utilizând instrucțiunea disp().

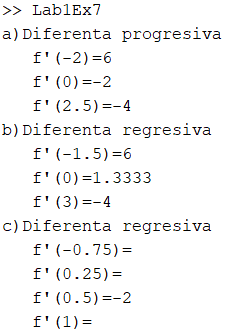


Fig 1.15 Executi punctului 1

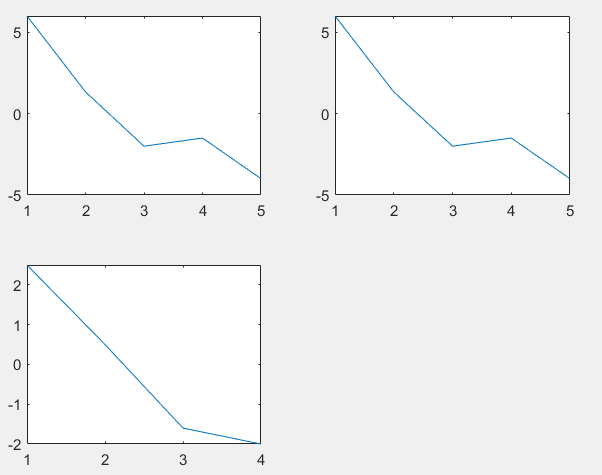


Fig 1.16 Graficele diferentialelor

Acest cod calculează derivata polinomului c folosind funcția polyder și apoi evaluează derivata la anumite valori de x folosind funcția polyval.Mai precis, dc = polyder(c) calculează derivata lui c, iar vx = [-2 -1.3 0.1 2.45] este un vector cu valorile x la care se evaluează derivata. Folosind funcția polyval, d = polyval(dc, vx) calculează valorile derivatei lui c la valorile specifice din vx.Ulterior, pentru a trasa graficul derivatei, se definește un vector x = -2:0.2:3, iar df = polyval(dc, x) evaluează derivata polinomului c în toate punctele din x. Apoi se trasează graficul folosind plot(x, df).În cele din urmă, pentru fiecare valoare de x din vx, se afișează valorile derivatei cu ajutorul unui ciclu for și a funcției disp.

|  |
| --- |
| dc=polyder(c);  vx=[-2 -1.3 0.1 2.45];  d=polyval(dc, vx);  x=-2:0.2:3;  df=polyval(dc, x);  plot(x, df);    for i=1:length(d)  disp(['d(' num2str(vx(i)) ')=' num2str(d(i))]);  end |

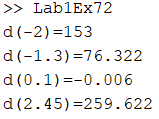


Fig 1.17 Derivatele în punctele x

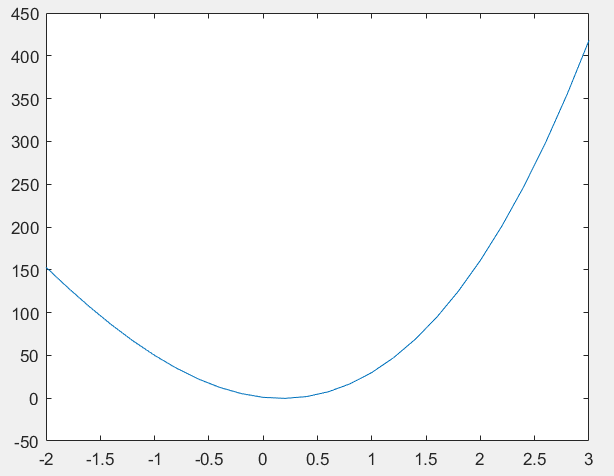


Fig 1.18 Graficul pe intervalul [-2, 3]

Ex 7 punctul 3

Acest cod calculează valoarea aproximativă a integralei definite a funcției date folosind metoda trapezelor.

Mai întâi, sunt definite vectorii x și y cu 11 elemente fiecare. Vectorul x este construit într-un mod specific prin atribuirea valorii (−1.1 + 0.1\*j) la fiecare element j (de la 1 la 11), iar vectorul y este calculat folosind aceleași valori din x și formula dată pentru fiecare element j.Funcția trapz este apoi folosită pentru a aproxima valoarea integralei definite a funcției y în raport cu x. Rezultatul este stocat în variabila Integ1. Această valoare este afișată la ieșirea din program folosind funcția disp.

|  |  |
| --- | --- |
| Cod | Execuție |
| x=1:11;  y=1:11;  for j=1:11  x(j) = (-1.1 + 0.1\*j);  y(j) = (((j \* x(j)^2)/((x(j)-1))) - (2 / (j+1)));  end  Integ1= trapz(x,y);  disp(Integ1); |  |

Ex 7 punctul 4

Acest cod defineste o functie matematica $f(x) = \frac{1}{\sin(x)+\cos(x)}$. Aceasta functie primeste ca argument un numar real $x$, aplica o formula matematica asupra sa si returneaza rezultatul. Functia poate fi folosita pentru a calcula valoarea functiei in orice punct dat, de exemplu apeland functia cu un anumit argument $x$, astfel: $f(x)$, si aceasta va returna valoarea lui $y$ calculata in functie de formula $ y=1./(sin(x)+cos(x))$.

Acest cod reprezintă o funcție numită f care primește un singur argument x și returnează valoarea funcției y = 1/(sin(x) + cos(x)).

Practic, dacă doriți să calculați valoarea funcției într-un anumit punct x, puteți apela funcția f(x). De exemplu, dacă ați dori să aflați valoarea funcției în punctul x = pi/4, ați putea apela funcția astfel: f(pi/4). Funcția va returna valoarea corespunzătoare a funcției în acel punct.

|  |  |
| --- | --- |
| Cod | Execuție |
| function y=f(x)  y=1./(sin(x)+cos(x));  end |  |

**Concluzie:**

În acest laborator au fost căpătate deprinderi în lucrul cu MATLAB, cu acest scop au fost îndeplinite un șir de sarcini, folosind instrumentele care le pune la dispoziția utilizatorului mediul MATLAB.

Primul lucru care a fost studiat în acest laborator este definirea vectorilor și matricilor, pe lîngă aceasta au fost efectuate operații cu aceste structuri de date. Un aspect foarte important al mediului MATLAB este faptul că acesta permite crearea unor fișiere cu extensia .m care permit o anumită parte a comenzilor care le scrim să le scrim deoparte și atunci cînd avem nevoie, să le apelăm doar indicînd denumirea fișierului în care acestea se află. De asemenea trebuie de menționat că MATLAB propune o multitudine de funcții care permit crearea graficilor unor funcții, aceste funcții ne permit să construim graficele cu setări proprii sau cu setări implicite.

În final putem spune că MATLAB este un instrument foarte puternic în lucrul cu calculele matematice și ne permite să ușurăm lucrul cu diverse tipuri de operații din matematică.

v=linspace(minim,maxim,număr\_de\_elemente) – se generează un vector linie v cu elementele, începând de la minim la maxim, cu pas constant şi având un număr de elemente egal cu număr\_de\_elemente.

Funcția "plot" este folosită pentru a trasa o curba continuă care trece prin toate punctele date. Aceasta poate fi utilizată pentru a reprezenta diverse tipuri de date, cum ar fi datele continue, semnalele și funcțiile matematice. De asemenea, "plot" poate fi folosit pentru a trasa mai multe curbe pe același grafic.

În schimb, funcția "stem" este folosită pentru a trasa o diagramă cu linii verticale care pornesc de la axa x și ajung până la punctele date de pe axa y. Această funcție este de obicei folosită pentru a reprezenta semnale discrete și date cu valori discrete.

Astfel, principala diferență dintre cele două funcții este modul în care datele sunt reprezentate grafic. "Plot" reprezintă datele ca o curba continuă, în timp ce "stem" le reprezintă ca linii verticale.