1. Prezentarea mediului de lucru Matlab

Acest prim laborator are drept scop prezentarea principalelor caracteristici ale mediului de lucru Matlab. Vor fi trecute în revistă, pe scurt, principiul de funcționare, cele mai folosite comenzi precum şi două instrumente foarte puternice ale acestuia: Simulink şi SpTool. De asemenea vor fi construite pas cu pas câteva scheme bloc simple pentru exemplificarea funcționalității acestuia.
Se poate spune despre Matlab că este un limbaj de nivel foarte înalt care prezintă performanțe crescute în ceea ce priveşte calculul tehnic. Pe lângă interpretorul de comenzi sunt prezente o mulțime de instrumente cum ar fi cele pentru vizualizarea datelor, pentru prelucrarea imaginilor şi sunetelor, pentru analiza circuitelor electrice, etc.

Spre deosebire de alte limbaje de nivel înalt, elementele de bază cu care se lucrează sunt vectorii. Utilizatorul poate defini şi folosi vectori ale căror dimensiuni inițiale nu trebuie specificate. Problemele legate de gestionarea memoriei la operațiile care presupun creşterea dimensiunii unui vector se fac automat, transparent utilizatorului, ceea ce este un avantaj față de limbaje de programare cum ar fi C sau C++.

Numele Matlab este o prescurtare a cuvintelor “matrix laboratory”. Aceasta deoarece, inițial programul a fost destinat pentru calculul cu matrici. Limbajul a evoluat şi a devenit un standard în universități când este vorba de cursuri introductive sau avansate de matematică sau inginerie. Funcțiile specifice unui anumit domeniu sunt grupate în colecții de funcții sau “toolboxes”. Acestea uşurează foarte mult folosirea programului în scop educațional sau de cercetare deoarece utilizatorul se poate concentra direct pe aplicarea unei serii de operații asupra unui set de date fără a se îngriji exclusiv de definirea acestor operații. Există mai multe colecții de funcții specifice domeniului electronicii cum ar fi cea pentru prelucrarea de semnale sau cea pentru domeniul comunicațiilor.
Utilizatorii isi pot implementa propriile functii. Pentru a putea fi folosite acestea trebuie sa se afle in directorul curent sau se adauga directorul in care se afla functiile folosind File-> Set Path.

Fereastra principală a programului permite accesul direct la interpreterul de comenzi. Acesta este un instrument care execută o secvență de cod linie cu linie. Secvența de cod poate fi introdusă direct de la tastatură, iar după fiecare linie se apasă tasta Enter sau poate fi scrisă într-un fişier de tip text, care se salvează cu extensia “.M” şi se execută prin simpla scriere a numelui fişierului. De asemenea liniile de cod scrise in editorul de tip text pot fi executate prin apasarea tastei F5 (se executa toate liniile scrise; comanda nu este valabila in cazul functiilor) sau a tastei F9 (se executa numai liniile de cod selectate).

Limbajul Matlab respectă principiile programării structurale, astfel că există o foarte mare asemănare între sintaxa şi structurile sale cu cea a limbajului C.

Exemplu

Să considerăm următoarea secvență de cod:

```
suma=0;
for i=1:10
     suma = suma + i;
     a(i) = i;
end
suma
a
plot(a,'k*')
```

Prin scrierea acesteia linie cu linie in fereastra principala sau in editorul de text se va obține următorul rezultat:

```
Suma=
55
a =
1  2  3  4  5  6  7  8  9  10
```

precum şi un grafic cu valorile vectorului “a”.

Sa se scrie in editorul de text exemplu de mai sus.

Obs:

1. In Matlab, spre deosebire de C, indexarea vectorilor se face incepand cu 1. Daca in exemplul de mai sus se scria `i=0:10` *atunci interpretorul ar fi generat eroare.*

2. Se poate afisa valoarea unei variabile prin simpla scriere a numelui acesteia sau a expresiei care o determina fara a adauga punct si virgula. Daca se scrie `Suma;` *acesta nu va determina afisarea valorii variabilei Suma.*

Pentru informatii privind orice functie sau comanda din Matlab se poate folosi comanda help sau doc

Exemplu :

```
help for sau doc for
help plot sau help plot
```

1.1 Tipuri de date in Matlab

Implicit toate opratiile in Matlab sunt realizate in virgula mobila si cu precizie dubla (64 biti). Exista insa posibilitatea utilizeze si date cu precizie simpla (32 de biti), intregi (8, 16 sau 32 de biti). Functiile care fac conversia de la un tip de date la altul sunt: double, single (functii pentru conversia la date de tip double si respectiv single), int8, int16, int32 (functii pentru conversia la date de tip intreg), unit8, unit16,uint32 (functii pentru conversia la date de tip intreg fara semn).

Exemplu:

```
a=1
b=single(a)
c=int8(a)
d=uint8(a)
```

1.2 Vectori si matrici in Matlab.

Un avantaj al Matlab-ului este usurinta cu care se lucreaza cu vectori si cu matrici. De altfel o secventa de cod care utilizeaza vectori se executa mai repede decat una care foloseste instructiuni gen `for`.

Exemplu:

Daca se doreste calcularea valorilor lui exp(-n) unde n=1..10, se poate scrie:

```
for n=1:10
     x(n)=exp(-n);
end
x
```

sau mai simplu

```
x=exp(-[1:10])
```

Exemple de sintaxe care folosesc vectori si matrici.

Sa se scrie exemplele de mai jos si sa se execute.

`a=[1 2 3]  b=[ 4 5 6]`	se definesc 2 vectori a si b de tip rand care au 3 elemente
`c=[1; 2; 3]`	se defineste un vector c de coloana care are 3 elemente
`A=[1 2 3; 4 5 6; 7 8 9]` `B=[2 3 4;5 6 7; 8 9 10]`	se definesc doua matrici A si B cu 3 randuri si 3 coloane *OBS: (Matlab-ul este key sensitive)*
`A'`	transpusa matricii A
`c=a'`	se transforma un vector de tip rand intr-un vector de tip coloana
`d=a*b'`	produsul scalar al vectorilor a si b
`e=a+b`	suma vectorilor a si b
`f=a.*b`	produsul vectorilor element cu element
`C=A*B`	produsul matricilor A si B
`D=A.*B`	produsul element cu element al matricilor A si B
`g=a.^2`	puterea a doua a fiecarui element a lui a
`E=A^2`	matricea A ridicata la puterea a doua
`F=A.^2`	puterea a doua a fiecarui element al matricii A
`x=[0:2:8]`	defineste un vector x ale carui valori sunt din 2 in 2 ([0 2 4 6 8])
`A(1,:)`	selecteaza primul rand al matricii A
`A(:,2)`	selecteaza a doua coloana a matricii A
`[a b]`	concateneaza vectorii a si b intr-un vector tip rand
`[a'; b']`	concateneaza vectorii a si b intr-un vector tip coloana
`[a; b]`	formeaza o matrice care are randurile vectorii a si respectiv b
`[a' b']`	formeaza o matrice care are coloanele vectorii a' si respectiv b'

Instructiuni de salvare, incarcare si stergere a variabilelor.

`save date.mat`	salveaza toate variabilele din workspace in fisierul date.mat. fisierul va fi salvat implicit in directorul curent.
`save date_partial.mat a, b, F`	salveaza variabile a, b, F in fisierul date_partial.mat
`clear all`	sterge toate variabilele.

`load date.mat`	incarca variabilele salvate in date.mat in workspace
`clear a b A`	sterge din workspace numai variabilele `a, b, A`

1.3 Afisarea si editarea graficelor in Matlab

Un alt avantaj al Matlab-ului este usurinta cu care pot afisa si edita graficele.
Mai jos se prezinta cateva exemple.

```
t=0:0.01:1;                % se defineste un vector de la 0 la 1 cu pasul de esantionare
                           de 0.01
x1=exp(-0.1*t);                %se defineste vectorul x1
x2=cos(2*pi*5*t);          % se defineste vectorul x2
```

A1. Care este valoarea maxima pe care o poate avea pasul de esantionare a semnalului x2?

```
x3=cos(2*pi*0.05*[1:100]);    %se defineste vectorul x3 (pasul de esantionare este
                               %Ts=1)
```

A2. Care este valoarea minima pe care o poate avea frecventa de esantionare a semnalului x3? Reprezentati semnalul x3 folosind o valoare a frecventei de esantionare care nu respecta teorema esantionarii.

```
figure(1);plot(t,x1);          % este activata figura 1 si se afiseaza semnalul x1
figure(2);plot(t,x2);grid on;  % este activata figura 2 si se afiseaza semnalul x2

figure(3);stem(x3);grid on;    % grid on determina aparitia gridului
xlabel('timp');                % returneaza textul care apare scris pe axa x
ylabel('amplitudine');         % returneaza textul care apare pe axa y
title('Semnalul x3');          % titlul figurii
axis([-10 150 -2 2]);          % se defineste axa x de la -10 la 150 si axa y de la -2
                               %la 2
```

Afisarea mai multor grafice pe aceeasi fereastra se realizeaza prin activarea proprietatii `hold on` a figurii curente.

```
figure(4); hold on; plot(t,x1);
grid on;
plot(t,x2,'r');
grid on;

%hold on este o proprietate a figurii 4 si odata executata determina afisarea urmatoarelor
%grafice tot in figura 4.
%Dezactivarea proprietatii se face executand hold off.
```

Impartirea unei ferestre in subferestre.

```
figure(5);
subplot(2,1,1);            %subplot determina impartirea figurii 5 in 2 subfiguri pe
                           randuri si activarea primei subfiguri
plot(t,x1,'-.');grid on;
xlabel('timp');                  % textul care apare scris pe axa x
ylabel('amplitudine x2');        % textul care apare scris pe axa y
title('Semnalul x2');            % titlul figurii

subplot(2,1,2);            %subplot determina impartirea figurii 5 in 2 subfiguri pe
                           %randuri si activarea primei subfiguri
stem(x2,'g.');grid on;
xlabel('timp');                  %  textul care apare scris pe axa x
ylabel('amplitudine x3');        % textul care apare pe axa y
title('Semnalul x3');            % titlul figurii
```

%Functia `subplot` determina impartirea impartirea figurii 5 in 2 subfiguri pe randuri. In prima se afiseaza semnalul x1 iar pe cea de-a doua x2.
%Prima cifra din `subplot` seteaza in cate randuri se imparte o figura iar cea de-a doua in %cate coloane. Ultima cifra seteaza a cata subfigura este activata.

Pentru editarea graficelor se pot folosi optiunile functiei `plot` sau mai simplu folosind “Show plot tools” toolbar.

1.4 Elemente de procesare a semnalelor in Matlab

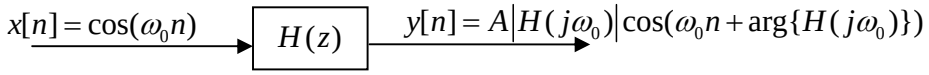
Pentru domeniul procesarii semnalelor Matlab-ul ofera un set complet functii grupate in Signal Processing Toolbox. De asemenea se folosesc functii care fac parte din Matlab.

```
abs([-1 -2 3 4])        %functia abs returneaza modulul unui vector
norm([1 2])             %functia norm returneaza norma L2 a unui vector

a=ones(1,10)            %functia ones returneaza un vector cu 10 elemente cu
                         valoarile 1
length(a)               %functia length returneaza numarul de elemente ale
                         vectorului a
stem(a)                  %functia stem afiseaza vectorul a folosind esantioane
b=zeros(1,10)     %functia zeros returneaza un vector cu 10 elemente,   b, cu
                  %valorile 0
figure(1);stem(b)
c=[1 zeros(1,10)]       %metoda prin care se poate simula δ[n]
figure(2);stem(c);

d=[1+1j*10 -2+1j*5 10+1j*5]      % returneaza un vector cu numere complexe
real(d)                          % returneaza partea reala  a unui numar
                                 %complex
imag(d)                          % returneaza partea imaginara a unui numar
                                 %complex
conj(d)                          % returneaza conjugatul unui numar complex
```

```
angle(d)                    % returneaza faza unui numarului unui
                            %complex
```

Raspunsul filtrelor numerice la semnale armonice.

Raspunsul unui filtru discret H(z) la un semnal armonic $x[n]$ este:



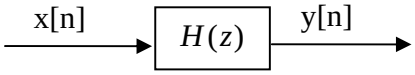
A3. Sa se determine raspunsul filtrului $H(z)=\frac{1}{1-0.5z^{-1}}$ la semnalul de intrare $x[n]=\cos(0.2\cdot\pi\cdot n)$.

Raspunsul filtrului in Matlab, poate fi determinat folosind functia `filter`.

```
%Exemplu de filtrare
x=cos(2*pi*0.1*[1:100]); % definarea semnalului x[n]=cos(0.2·π·n).
num=[1 0];
den=[1 -0.5];
y=filter(num,den,x); %Functia filter returneaza raspunsul filtrului definit prin
                     %numaratorul (num) si numitorul (den) functiei de transfer, la
                     %semnalul x.
```

A4. Sa se afiseze pe aceelasi grafic semnalul x[n], semnalul calculat la aplicatia A3 si semnalul y[n] returnat de functia `filter`.

Raspunsul tranzitoriu al filtrelor numerice.



Daca $H[z]$este functia de transfer a filtrului numeric iar $h[n]$ raspunsul sau la impuls, relatia intre intrarea si iesirea filtrului este:

$$y[n]=x[n]*h[n] \quad \text{sau} \quad Y(z)=X[z]\cdot H(z)$$

unde "$*$" reprezinta produsul de convolutie definit prin:

$$x[n]*h[n]=\sum_{k=-\infty}^{\infty}x[k]\cdot h[n-k]$$

A5. Sa se determine raspunsul filtrului $H(z)=1+z^{-2}$ la semnalul de intrare $x[n]=\delta[n]-\delta[n-1]$.

Produsul de convolutie se calculeaza folosind functia Matlab `conv` sau `filter`. Deoarece vectorii, ce constitue parametrii de intrare a functiei `conv`, au un numar limitat de elemente functia se poate folosi pentru a determina raspunsul filtrelor de tip FIR (filtre cu raspuns finit la impuls).

```
x=[1 -1];    % definirea semnalului x[n]=δ[n]-δ[n-1]
h=[1 0 1];   % definirea raspunsului la impuls al filtrului
```

```
y=conv(x,h);  % functia conv returneaza produsul de convolutie dintre semnalele
%x[n] si h[n]
```

A6. Sa se verifice in Matlab rezultatul afisand rezultatul obtinut la aplicatia A5 si semnalul y[n] calculat in Matlab.

A7. Sa se determine raspunsul filtrului $H(z)=\frac{1}{1-z^{-1}}$ la semnalul de intrare $x[n]=\sigma[n]-\sigma[n-8]$ ($\sigma[n]$ reprezinta functia treapta unitate).

Deoarece raspunsul la impuls al filtrului are un numar infinit de elemente, in Matlab se poate folosi functia `filter` a carei parametri au fost explicati mai sus.

A8. Sa se defineasca in Matlab semnalul $x[n]$ si sa se foloseasca functia `filter` pentru a determina raspunsul, y[n], al filtrului la semnalul x[n]. Sa se afiseze in aceeasi figura, semnalul calculat la aplicatia A7 si semnalul y[n] obtinut in Matlab.

Elemente de analiza a filtrelor numerice in Matlab.

Cum sa vazut in aplicatiile anterioare, pentru a defini un filtru in Matlab este suficient sa se scrie in 2 vectori coeficentii polinoamelor corespunzatoare numaratorului si numitorului (primul element al vectorului corespunde puterii celei mai mari a fiecarui polinom). De exemplu, filtrul numeric a carui functie de transfer este:

$$H(z)=\frac{1-0.1z^{-1}}{1+0.5z^{-1}+0.06z^{-2}} \text{ este definit prin vectorii } \begin{cases} num=[1\ -0.1\ 0]; \\ den=[1\ 0.5\ 0.06]; \end{cases}$$

iar filtrul numeric a carui functie de transfer este:

$$H(z)=\frac{0.1z^{-1}}{1+0.5z^{-1}+0.06z^{-3}} \text{ este definit prin vectorii } \begin{cases} num=[0\ 0.1\ 0\ 0]; \\ den=[1\ 0.5\ 0\ 0.06]; \end{cases}$$

Raspunsul la impuls al filtrului se poate determina in Matlab folosind functia `impz`.

```
figure(1);
hz=impz(num,den,N); %num si den reprezinta numaratorul si numitorul functie de transfer
                    %a filtrului numeric, iar N numarul de esantioane care se doresc afisate
```

A9. Sa se calculeze raspunsul la impuls al filtrului $H(z)=\frac{1-z^{-1}}{1+0.5\cdot z^{-1}+0.06\cdot z^{-2}}$ si sa afiseze in Matlab raspunsul la impuls calculat si cel determinat functia Matlab `impz`.

Caracteristica de frecventa a filtrelor numerice este:

$$H(\omega)=H(z)\big|_{z=e^{j\omega}}=\sum_{n=-\infty}^{\infty}h[n]\cdot e^{-j\omega n}$$

Se poate demonstra ca funtia $H(\omega)$ este periodica cu perioada 2π. In Matlab pentru a determina se poate folosi functia `freqz` care are ca parametri numaratorul si numitorul functie de transfer si valorile in care se calculeaza $H(\omega)$.

```
w=[-pi:2*pi/512:pi-2*pi/512];  % intervalul de frecventa este definit intre -pi si pi in
                               %512 puncte
h=freqz(num,den,w);            % returneaza caracteristicile de frecventa si faza la
                               %frecventele indicate de w
```

Urmatorul script afiseaza caracteristica de amplitudine si de faza a unui filtru numeric.

```
figure(2);
subplot(2,1,1);plot(w,abs(h));grid on;   %Caracteristica de amplitudine
xlabel('Frecventa');
ylabel('Amplitudine');
subplot(2,1,2);plot(w,angle(h));grid on; %Caracteristica de faza
xlabel('Frecventa');
ylabel('Faza');
```

Obs: Functia `freqz` poate fi utilizata si in forma `[h,w]=freqz(num,den).` In acesta situatie, vectorul corespunzator frecventelor este generat de functie.

Diagrama poli-zerouri se determina in Matlab folosind functia tf2zp si se afiseaza folosind functia zplane.

A10. Sa se calculeze polii si zerourile functiei de transfer si sa se verifice rezultatul obtinut folosind functia tf2zp.

$$H(z)=\frac{1-0.9z^{-1}}{1+0.5z^{-1}+0.2z^{-2}}$$

```
[z,p,k]=tf2zp(num,den);   %returneaza zerourile si polii functiei de transfer
zplane(z,p);              %afiseaza pozitia polilor si a zerourilor.
```

Calcularea si afisarea spectrului unui semnal in Matlab

Spectrul unui semnal x[n] este data de transformarea Fourier in timp discret:

$$X(\omega)=\sum_{n=-\infty}^{\infty}x[n]\cdot e^{-j\omega n}$$

A11. Sa se arate ca functia $X(\omega)$ este o functie periodica cu perioada 2π.

In Matlab, spectrul unui semnal calculeaza folosind Transformata Fourier Discreta:

$$X[k]=\sum_{n=0}^{N-1}x[n]e^{-j\frac{2\pi}{N}nk},k=0..N-1$$

In scriptul urmator se genereaza un semnal f[n], se determina spectrul semnalului si se afiseaza modulul spectrului.

```
f=cos(2*pi*0.1*[1:50]+pi/4);
F=fft(f);                      % Transformata Fourier discreta a semnalului f[n]
figure(4);
subplot(4,1,1);stem(abs(F));grid on;
```

```
subplot(4,1,2);stem(fftshift(abs(F)));grid on; %functia fftshift
```
%transleaza componentele de joasa frecventa ale spectrului pe centrul axei.

A12. Modificati stem(fftshift(abs(F))) astfel incat abcisa subfigurii (4,1,2) sa varieze intre [-pi,pi). Afisati rezultatul in subfigura (4,1,3).

```
subplot(4,1,3);hold on;stem(ifft(F));stem(f,'r*');grid on;
```
%Transformata Fourier discreta inversa a semnalului f[n].