

# 1. INTRODUCERE ÎN MATLAB

## 1.1. Introducere

MATLAB este un pachet de programe dedicat calcului numeric și reprezentărilor grafice. Elementul de bază cu care operează este matricea, de aici provenind și numele său: MATrix LABoratory. Resursele sale de calcul și reprezentare grafică sunt bogate, permițând operații matematice fundamentale, analiza datelor, programare, reprezentări grafice 2D și 3D, realizarea de interfețe grafice etc. Din punct de vedere al construcției sale, MATLAB este alcătuit dintr-un nucleu de bază în jurul căruia sunt grupate TOOLBOX-urile. Acestea reprezintă niște aplicații specifice, fiind de fapt colecții extinse de funcții MATLAB care dezvoltă mediul de programare de la o versiune la alta, pentru a rezolva probleme din diverse domenii. În prelucrarea numerică a semnalelor cel mai des utilizat este toolbox-ul SIGNAL PROCESSING.

## 1.2. Desfășurarea lucrării

MATLAB operează cu două tipuri de ferestre:

1. fereastra de comenzi;
2. fereastra pentru reprezentări grafice.

La deschiderea programului MATLAB va apărea pe ecranul calculatorului fereastra de comenzi, având în partea de sus bara de meniuri aferentă. Simbolul “>>” reprezintă prompterul MATLAB și se află la începutul fiecărei linii de comandă din fereastra de comenzi. Ferestrele pentru reprezentări grafice vor apărea atunci când se va cere prin comenzi specifice afișarea unor grafice.

***Atenție:** Comenzile introduse anterior pot fi readuse în linia de comandă prin folosirea săgeților ↑ și ↓ (căutarea se face ca într-o “listă”).*

# 1. INTRODUCERE ÎN MATLAB

---

## 1.2.1. Funcții MATLAB de interes general

- **help** – furnizează informații despre MATLAB și funcțiile acestuia.

*Sintaxă:*

`help nume` – furnizează informații despre nume (poate fi un nume de funcție sau un nume de director).

*Exemplu:*

`help fft` – furnizează informații despre transformata Fourier discretă.

- **who** – listează numele variabilelor din spațiul de lucru.
- **whos** – furnizează informații suplimentare referitoare la variabilele din spațiul de lucru (nume, dimensiune etc.).
- **format** – stabilește formatul extern de afișare al numerelor pe ecran.

*Sintaxa:*

`format opțiune` – parametrul `opțiune` poate fi:

- `short` – 5 cifre // formatul implicit
- `long` – 15 cifre
- `short e` – 5 cifre + exp (puteri ale lui 10)
- `long e` – 15 cifre + exp (puteri ale lui 10)
- `hex` – hexazecimal
- `bank` – exprimare cu 2 zecimale
- `rat` – exprimare rațională
- etc.

Pentru mai multe informații tastați `help format`.

*Exemple:*

`format short`

`x=pi` → `x =`  
3.1416

`format long`

`x` → `x =`  
3.14159265358979

Să se verifice și celelalte tipuri de format folosind aceeași valoare `x`.

- **clear** – șterge din memorie una sau mai multe variabile.

*Sintaxe:*

`clear v` – șterge din memorie variabila `v`.

`clear v1 v2` – șterge din memorie variabilele `v1` și `v2` (sintaxa e valabilă pentru oricâte variabile).

`clear` – șterge din memorie toate variabilele definite până în acel moment.

## 1. INTRODUCERE ÎN MATLAB

---

- **lookfor** – listează toate numele de fișiere care au în prima linie a help-ului cuvintele menționate ca argument, precum și prima linie din help.

*Sintaxă:*

`lookfor cuvânt` – listează toate numele de fișiere care conțin în prima linie a help-ului `cuvânt`, precum și prima linie din help.

*Exemplu:*

`lookfor ifft` – listează toate numele de fișiere care conțin în prima linie a help-ului `ifft`, precum și prima linie din help.

- **dir** – afișează numele tuturor fișierelor din directorul curent sau din orice alt director precizat ca argument.

*Sintaxe:*

`dir` – afișează numele tuturor fișierelor din directorul curent.

`dir nume` – afișează numele tuturor fișierelor din directorul `nume`.

- **cd** – returnează numele directorului curent sau schimbă directorul de lucru.

*Sintaxe:*

`cd` – returnează numele directorului curent.

`cd c:\matlab\nume_director` – schimbă directorul de lucru în `nume_director` (presupunând că MATLAB este instalat pe `c:\`).

**Atenție:** *Un anumit program MATLAB aflat într-un anumit director nu poate fi rulat decât dacă directorul respectiv este directorul de lucru.*

### 1.2.2. Variabile și constante speciale în MATLAB

- **ans** – variabilă creată automat în care este returnat rezultatul unui calcul, atunci când expresia nu a avut asignat un nume.

*Exemplu (se va tasta direct în fereastra de comenzi):*

3 → ans = 3 // nu s-a alocat nici un nume

x=2 → x = 2 // s-a alocat numele x

- **pi** – valoarea  $\pi$ .
- **Inf** – reprezentarea lui  $+\infty$ .

*Exemplu:*

y=1/0 → Warning: Divide by zero  
y =  
Inf //avertizează asupra împărțirii la 0

## 1. INTRODUCERE ÎN MATLAB

---

- **NaN** – reprezentarea lui 0/0 sau  $\infty/\infty$ .

*Exemplu:*

```
z=0/0 → Warning: Divide by zero
        y =
           NaN           //avertizează asupra împărțirii la 0
```

- **i** sau **j** – folosite reprezentarea numerelor complexe.

*Exemplu:*

```
a=2+4*i → a=
           2.0000+4.0000i
```

### **Atenție:**

- dacă după o linie de comandă urmează semnul “ ; ” atunci rezultatul nu va mai fi afișat (excepție fac comenzile grafice);
- dacă în fața unei linii de comandă se pune semnul “ % ” atunci se face abstracție de linia respectivă (este interpretată ca o linie de comentariu);
- dacă se dorește continuarea unei instrucțiuni pe linia următoare se folosesc “...” urmate de enter;

Verificați următoarele *exemple* (se va tasta direct în fereastra de comenzi):

```
p=pi ;           // nu se va afișa valoarea p (dar există în memorie)
q=pi/2          // va afișa valoarea q
%r=pi/4         // nu se ia în considerare această linie
v=r/2           // va rezulta o eroare deoarece nu îl cunoaște pe r
s=1+2+3... enter
4+5+6           // instrucțiunea se continuă și pe linia următoare
```

### 1.2.3. Matricea – element de bază în MATLAB

MATLAB lucrează numai cu un singur tip de obiecte și anume matrice numerice, având elemente reale sau complexe. Astfel scalarii sunt priviți ca matrice de dimensiune 1 x 1, iar vectorii ca matrice de dimensiune 1 x n (dacă este vector linie) sau n x 1 (dacă este vector coloană).

Reguli privind modul de definire a matricelor:

- elementele matricei sunt cuprinse între paranteze drepte [ ]
- elementele unei linii se separă prin pauză (blanc) sau virgulă
- liniile matricei se separă prin ; sau *enter*.

*Exemplu:* Fie matricea  $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$  și vectorii  $B = (7 \ 8 \ 9)$ ,  $C = \begin{pmatrix} -1 \\ -2 \end{pmatrix}$

## 1. INTRODUCERE ÎN MATLAB

---

$A=[1, 2, 3; 4, 5, 6]$  →  $A =$   
1 2 3  
4 5 6

$A=[1 2 3; 4 5 6]$  →  $A =$   
1 2 3  
4 5 6

$A=[1 2 3$  *enter*  
 $4 5 6]$  →  $A =$   
1 2 3  
4 5 6

$B=[7 8 9]$  →  $B =$   
7 8 9

$C=[-1; -2]$  →  $C =$   
-1  
-2

Pentru o matrice M:

- $M(i, j)$  reprezintă elementul din matricea M corespunzător liniei  $i$  și coloanei  $j$
- $M(i)$  reprezintă elementul  $i$  din matrice, numărarea elementelor făcându-se pe coloane.

Pentru un vector v:

- $v(i)$  reprezintă elementul de pe poziția  $i$  din vector.

*Exemplu:* Pentru A, B, C definite anterior verificați:

$A(2, 3)$  → ans =  
6

$A(4)$  → ans =  
5

$B(2)$  → ans =  
8

$C(2)$  → ans =  
-2

Dacă dorim să schimbăm elementele unei matrice sau să mai adăugăm alte elemente, fără a mai rescrie întreaga matrice, procedăm ca în exemplul următor:

$A(2, 3)=0$  →  $A =$   
1 2 3  
4 5 0

$A(3, 3)=-3$  →  $A =$   
1 2 3  
4 5 0  
0 0 -3

## 1. INTRODUCERE ÎN MATLAB

---

$$C(3) = 6 \quad \rightarrow \quad C = \begin{matrix} -1 \\ -2 \\ 6 \end{matrix}$$

Se pot construi matrice de dimensiuni mai mari pornind de la matrice de dimensiuni mai reduse. Pentru exemplificare vom folosi matricea A și vectorii B și C în ultima lor formă (A – 3 x 3, B – 1 x 3, C – 3 x 1):

$$D = [A; B] \quad \rightarrow \quad D = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 0 \\ 0 & 0 & -3 \\ 7 & 8 & 9 \end{matrix}$$

// s-a construit matricea D de dimensiune 4 x 3, prin adăugarea vectorului B la matricea A (ca ultimă linie).

**Atenție:** A și B au același număr de coloane (3) pentru a fi posibilă construcția.

$$E = [A, C] \quad \rightarrow \quad E = \begin{matrix} 1 & 2 & 3 & -1 \\ 4 & 5 & 0 & -2 \\ 0 & 0 & -3 & 6 \end{matrix}$$

// s-a construit matricea E de dimensiune 3 x 4, prin adăugarea vectorului C la matricea A (ca ultimă coloană).

**Atenție:** A și C au același număr de linii (3) pentru a fi posibilă construcția.

Dacă  $v$  este un *vector* atunci:

- $v(i:k)$  – selectează elementele de pe pozițiile  $i, i + 1, i + 2, \dots, k$  ale vectorului  $v$ ; dacă  $i > k$  atunci vectorul rezultat este gol (nu are nici un element).
- $v(i:j:k)$  – selectează elementele de pe pozițiile  $i, i + j, i + 2j, \dots$  până la  $k$ , ale vectorului  $v$  (selectează cu pasul  $j$ ); dacă  $j > 0$  și  $i > k$  sau  $j < 0$  și  $i < k$  atunci vectorul rezultat este gol.
- $v([i, j, k])$  – selectează elementele de pe pozițiile  $i, j$  și  $k$ .
- $v(:)$  – dacă vectorul este linie atunci el devine coloană; dacă vectorul este coloană atunci el rămâne nemodificat.

Dacă  $M$  este o *matrice* atunci:

- $M(:, j)$  – selectează coloana  $j$  a matricei  $M$ .
- $M(i, :)$  – selectează linia  $i$  a matricei  $M$ .
- $M(:, i:j)$  – selectează coloanele de la  $i$  la  $j$  ale matricei  $M$ .
- $M(i:j, :)$  – selectează liniile de la  $i$  la  $j$  ale matricei  $M$ .

## 1. INTRODUCERE ÎN MATLAB

---

- $M(:, i:j:k)$  – selectează coloanele  $i, i + j, i + 2j, \dots$  până la  $k$  ale matricei  $M$  (selectează cu pasul  $j$ ).
- $M(i:j:k, :)$  – selectează liniile  $i, i + j, i + 2j, \dots$  până la  $k$  ale matricei  $M$ .
- $M(i:j, k:l)$  – extrage submatricea formată cu elementele aflate la intersecția liniilor de la  $i$  la  $j$  și coloanelor de la  $k$  la  $l$  ale matricei  $M$ .
- $M(:, [i, j, k])$  – selectează coloanele  $i, j$  și  $k$  ale matricei  $M$ .
- $M([i, j, k], :)$  – selectează liniile  $i, j$  și  $k$  ale matricei  $M$ .
- $M([i, j, k], [l, m, n])$  – extrage submatricea formată cu elementele aflate la intersecția liniilor  $i, j$  și  $k$  și coloanelor  $l, m$  și  $n$  ale matricei  $M$ .
- $M(:, :)$  – selectează întreaga matrice  $M$ .
- $M(i:j)$  – selectează elementele de  $i$  la  $j$  ale matricei  $M$  și le pune sub forma unui vector linie (elementele într-o matrice se numără pe coloane).
- $M(:)$  – selectează toate elementele matricei  $M$  și le pune sub forma unui vector coloană (pune coloanele matricei  $M$  una sub alta, sub forma unui vector coloană).

*Verificați sintaxele de mai sus folosind matricea  $A$  și vectorii  $B$  și  $C$  din exemplele anterioare sau construind alte matrice și vectori.*

### 1.2.4. Vectori și matrice uzuale

- **Generarea vectorilor cu pas liniar**

*Sintaxe:*

- $v = \text{inițial} : \text{pas} : \text{final}$  – se generează un vector linie  $v$  cu elementele începând de la  $\text{inițial}$  la  $\text{final}$ , cu pasul egal cu  $\text{pas}$  (pasul poate fi și negativ dar atunci valoarea inițială trebuie să fie mai mare decât valoarea finală).
- $v = \text{inițial} : \text{final}$  – se generează un vector linie  $v$  cu elementele începând de la  $\text{inițial}$  la  $\text{final}$ , cu pasul egal cu 1.
- $v = \text{linspace}(\text{minim}, \text{maxim}, \text{număr\_de\_elemente})$  – se generează un vector linie  $v$  cu elementele începând de la  $\text{minim}$  la  $\text{maxim}$ , cu pas constant și având un număr de elemente egal cu  $\text{număr\_de\_elemente}$ .

Verificați următoarele *exemple*:

```
v=3:7:40
```

```
u=5:10
```

```
d=17:-3:4
```

```
l=linspace(17,58,4)
```

```
p=linspace(-pi,pi,6)
```

```
q=linspace(pi,-pi,6)
```

## 1. INTRODUCERE ÎN MATLAB

---

- **Generarea vectorilor cu pas logaritmic**

*Sintaxe:*

- `v=logspace(minim,maxim)` – se generează un vector linie `v` având 50 de elemente distribuite logaritmic între  $10^{\text{minim}}$  și  $10^{\text{maxim}}$ .
- `v=logspace(minim,maxim,număr_de_elemente)` – se generează un vector linie `v` având `număr_de_elemente` elemente distribuite logaritmic între  $10^{\text{minim}}$  și  $10^{\text{maxim}}$ .

**Atenție:** Dacă `maxim=pi` atunci elementele vor fi distribuite logaritmic între  $10^{\text{minim}}$  și  $\pi$ .

Verificați următoarele *exemple*:

```
g=logspace(1,2)
r=logspace(1,pi)
h=logspace(1,2,5)
k=logspace(0,pi,5)
```

- **Matricea goală**

*Sintaxa:*

- `x=[]` – generează o matrice goală (fără nici un element)

*Exemplu:*

```
x=[]      →      x =
                [ ]
```

- **ones - Matricea unitate**

*Sintaxe:*

- `ones(n)` – returnează o matrice de dimensiune `n x n` cu toate elementele egale cu 1.
- `ones(m,n)` – returnează o matrice de dimensiune `m x n` cu toate elementele egale cu 1.
- `ones(size(M))` – returnează o matrice de dimensiunea matricei `M` cu toate elementele egale cu 1.

Verificați următoarele *exemple*:

```
ones(3)
ones(3,3)
ones(1,5)
ones(5,1)
ones(3,2)
ones(size(D)) // unde D este matricea definită anterior (vezi pag. 6)
```



## 1. INTRODUCERE ÎN MATLAB

---

- **zeros - Matricea zero**

*Sintaxe:*

- `zeros(n)` – returnează o matrice de dimensiune  $n \times n$  cu toate elementele egale cu 0.
- `zeros(m,n)` – returnează o matrice de dimensiune  $m \times n$  cu toate elementele egale cu 0.
- `zeros(size(M))` – returnează o matrice de dimensiunea matricei  $M$  cu toate elementele egale cu 0.

Verificați următoarele *exemple*:

```
zeros(3)
zeros(3,3)
zeros(1,5)
zeros(5,1)
zeros(3,2)
zeros(size(D)) // unde D este matricea definită anterior (vezi pag. 6)
```

- **eye - Matricea identitate**

*Sintaxe:*

- `eye(n)` – returnează o matrice identitate de dimensiune  $n \times n$ .
- `eye(m,n)` – returnează o matrice de dimensiune  $m \times n$  având elementele primei diagonale egale cu 1 iar restul elementelor egale cu 0.
- `eye(size(M))` – returnează o matrice de dimensiune egală cu dimensiunea matricei  $M$ , având elementele primei diagonale egale cu 1 iar restul elementelor egale cu 0.

Verificați următoarele *exemple*:

```
eye(3)
eye(3,3)
eye(1,5)
eye(5,1)
eye(3,2)
eye(size(D)) // unde D este matricea definită anterior (vezi pag. 6)
```

- **rand - Matricea cu numere aleatoare cu distribuție uniformă**

*Sintaxe:*

- `rand(n)` – returnează o matrice de dimensiune  $n \times n$  având drept elemente numere aleatoare cu distribuție uniformă între 0 și 1.
- `rand(m,n)` – returnează o matrice de dimensiune  $m \times n$  având drept elemente numere aleatoare cu distribuție uniformă între 0 și 1.
- `rand(size(M))` – returnează o matrice de dimensiunea matricei  $M$  având drept elemente numere aleatoare cu distribuție uniformă între 0 și 1.

## 1. INTRODUCERE ÎN MATLAB

---

Verificați următoarele *exemple*:

```
rand(3)
rand(3,3)
rand(1,5)
rand(5,1)
rand(3,2)
rand(size(D)) // unde D este matricea definită anterior (vezi pag. 6)
```

- **randn** - *Matricea cu numere aleatoare cu distribuție normală (gaussiană)*

*Sintaxe:*

- `randn(n)` – returnează o matrice de dimensiune  $n \times n$  având drept elemente numere aleatoare cu distribuție normală (gaussiană) de medie nulă și varianța unitară.
- `randn(m,n)` – returnează o matrice de dimensiune  $m \times n$  având drept elemente numere aleatoare cu distribuție normală de medie nulă și varianța unitară.
- `randn(size(M))` – returnează o matrice de dimensiunea matricei  $M$  având drept elemente numere aleatoare cu distribuție normală (gaussiană) de medie nulă și varianța unitară.

Verificați următoarele *exemple*:

```
randn(3)
randn(3,3)
randn(1,5)
randn(5,1)
randn(3,2)
randn(size(D)) // unde D este matricea definită anterior (vezi pag. 6)
```

- **diag** - *Matricea diagonală*

*Sintaxe:*

Dacă  $v$  este un *vector* (linie sau coloană) atunci

- `diag(v)` – returnează o matrice pătrată diagonală, cu elementele vectorului  $v$  pe diagonala principală.
- `diag(v,k)` – returnează o matrice pătrată cu elementele vectorului  $v$  pe diagonala  $k$  deasupra celei principale, dacă  $k > 0$ , sau sub cea principală dacă  $k < 0$ ; restul elementelor sunt 0.

Dacă  $M$  este o *matrice* atunci

- `diag(M)` – returnează un vector coloană ce conține elementele de pe diagonala principală a matricei  $M$ .
- `diag(M,k)` – returnează un vector coloană ce conține elementele din matricea  $M$  de pe diagonala  $k$  deasupra celei principale, dacă  $k > 0$ , sau sub cea principală, dacă  $k < 0$ .

## 1. INTRODUCERE ÎN MATLAB

---

Se va defini un vector linie  $a$  și o matrice  $A$ :

```
a=randn(1,5)
```

```
A=randn(5)
```

Verificați următoarele *exemple*:

```
diag(a)
```

```
diag(a,1)
```

```
diag(a,-1)
```

```
diag(a,2)
```

```
diag(a,-2)
```

```
diag(A)
```

```
diag(A,2)
```

```
diag(A,-2)
```

```
diag(diag(A))
```

- **tril** - *Matricea inferior triunghiulară*

*Sintaxe:*

- `tril(M)` – extrage matricea inferior triunghiulară din matricea  $M$  (anulează toate elementele matricei  $M$  de deasupra diagonalei principale).
- `tril(M,k)` – înlocuiește cu 0 toate elementele de deasupra diagonalei  $k$  din matricea  $M$  (raportarea se face la diagonala principală – vezi sintaxa de la `diag`).

Pentru matricea  $A$  definită anterior verificați următoarele *exemple*:

```
tril(A)
```

```
tril(A,1)
```

```
tril(A,-1)
```

- **triu** - *Matricea superior triunghiulară*

*Sintaxe:*

- `triu(M)` – extrage matricea superior triunghiulară din matricea  $M$  (anulează toate elementele matricei  $M$  de sub diagonala principală)
- `triu(M,k)` – înlocuiește cu 0 toate elementele de sub diagonala  $k$  din matricea  $M$  (raportarea se face la diagonala principală – vezi sintaxa de la `diag`).

Pentru matricea  $A$  definită anterior verificați următoarele *exemple*:

```
triu(A)
```

```
triu(A,1)
```

```
triu(A,-1)
```

## 1. INTRODUCERE ÎN MATLAB

---

### 1.2.5. Dimensiunea unei matrice. Determinantul și inversa unei matrice

- **length, size** – *Determinarea dimensiunii variabilelor*

*Sintaxe:*

Dacă  $v$  este un vector și  $M$  este o matrice  $m \times n$  atunci

- `length(v)` – returnează numărul de elemente (lungimea) vectorului  $v$ .
- `length(M)` – returnează maximul dintre numărul de linii și numărul de coloane al matricei  $M$  (maximul dintre  $m$  și  $n$ ).
- `[l, c]=size(M)` – returnează numărul de linii ( $l$ ) și numărul de coloane ( $c$ ) pentru matricea  $M$ .
- `[l, c]=size(v)` – în acest caz una dintre dimensiuni va fi egală cu 1; dacă  $v$  este un vector linie atunci  $l = 1$ , iar dacă este coloană atunci  $c = 1$ .

Se va defini un vector linie  $a$ , un vector coloană  $b$  și o matrice  $C$ :

```
a=randn(1,5)
```

```
b=randn(5,1)
```

```
C=randn(3,4)
```

Verificați următoarele *exemple*:

```
length(a)
```

```
length(b)
```

```
length(C)
```

```
size(a)
```

```
size(b)
```

```
size(C)
```

- **det** – *Determinantul unei matrice*

*Sintaxă:*

Dacă  $M$  este o matrice pătratică (numărul de linii = numărul de coloane) atunci

- `det(M)` – calculează determinantul matricei  $M$ .

Se vor defini două matrice:

```
M=randn(4)
```

```
N=randn(4,3)
```

Verificați următoarele *exemple*:

```
det(M)
```

```
det(N)
```

- **inv** – *Inversa unei matrice*

*Sintaxă:*

Dacă  $M$  este o matrice pătratică cu determinantul diferit de zero atunci

- `inv(M)` – calculează inversa matricei  $M$ .

Verificați folosind matricele  $M$  și  $N$  definite la punctul precedent.

## 1. INTRODUCERE ÎN MATLAB

---

### 1.2.6. Instrucțiuni de control logic. Operatori relaționali și operatori logici

- **if, else, elseif** – *Execuția condiționată*

*Sintaxe:*

- if expresie  
    instrucțiuni  
end  
// dacă expresie este adevărată se execută instrucțiuni;  
   dacă expresie este falsă se trece după end.
- if expresie  
    instrucțiuni\_1  
else  
    instrucțiuni\_2  
end  
// dacă expresie este adevărată se execută instrucțiuni\_1;  
   dacă expresie este falsă se execută instrucțiuni\_2.
- if expresie\_1  
    instrucțiuni\_1  
elseif expresie\_2  
    instrucțiuni\_2  
end  
// dacă expresie\_1 este adevărată se execută instrucțiuni\_1;  
   dacă expresie\_1 este falsă și expresie\_2 este adevărată se execută  
   instrucțiuni\_2.

- **for** – *Repetarea unui număr de instrucțiuni de un anumit număr de ori*

*Sintaxă:*

- for index=inițial:pas:final  
    instrucțiuni  
end  
// pentru index parcurgând intervalul de la inițial la final cu pasul  
   pas se execută instrucțiuni.

- **while** – *Repetarea unui număr de instrucțiuni atâta timp cât o condiție specificată este adevărată*

*Sintaxă:*

- while expresie  
    instrucțiuni  
end  
// cât timp expresie este adevărată se execută instrucțiuni.

## 1. INTRODUCERE ÎN MATLAB

---

- **Operatori relaționali**

- < - mai mic
- <= - mai mic sau egal
- > - mai mare
- >= - mai mare sau egal
- == - identic
- ~= - diferit

- **Operatori logici**

- & - operatorul ȘI logic
- | - operatorul SAU logic
- ~ - operatorul NU logic

### 1.2.7. Crearea programelor MATLAB (crearea fișierelor de comenzi \*.m)

Pentru secvențe lungi de comenzi se recomandă crearea și lansarea în execuție a unui program MATLAB. Acesta este un fișier “text” având extensia .m și conținând succesiunea dorită de comenzi MATLAB. După creare, fișierul devine o nouă comandă externă MATLAB, care poate fi lansată în execuție prin simpla introducere de la tastatură a numelui fișierului (fără extensia .m).

Pentru crearea unui astfel de fișier se parcurg următorii pași:

- în bara de meniuri a ferestrei de comenzi se selectează File, iar în interiorul acesteia New, urmată de M-file. Se va deschide astfel o nouă fereastră (sesiune de editare cu editorul NOTEPAD) cu bară de meniuri proprie și cu numele Untitled. În acest fișier se scrie programul MATLAB dorit.
- pentru a salva fișierul astfel creat sub un alt nume se selectează din bara de meniuri a noii ferestre comanda File, urmată de Save As...; va apărea o nouă fereastră de dialog în care vom preciza numele fișierului (File name:) însoțit de extensia .m și locul unde dorim să îl salvăm (Save in:).

**Atenție:**

- pentru a rula un program MATLAB trebuie ca directorul în care a fost salvat să fie directorul de lucru (vezi comanda cd, pagina 3).
- pentru a fi luate în considerare eventualele modificări făcute într-un program MATLAB, înainte de o nouă rulare, fișierul trebuie salvat (File, urmat de Save).

**E1. Exercițiu:**

Folosind sintaxele și indicațiile din secțiunile 1.2.6. și 1.2.7. realizați un program MATLAB care să citească elementele unui vector cu elemente aleatoare cu distribuție normală (gaussiană) și să afișeze fiecare element negativ întâlnit.

## 1. INTRODUCERE ÎN MATLAB

---

### 1.2.8. Crearea funcțiilor MATLAB (crearea fișierelor funcție)

Dacă prima linie a unui fișier MATLAB (\*.m) conține la început cuvântul `function` atunci fișierul respectiv e declarat ca fișier funcție. Aceste fișiere pot fi adăugate ca funcții noi în structura MATLAB. Forma generală a primei linii a unui fișier funcție este:

```
function [parametrii_ieșire]=nume(parametrii_intrare)
```

cu următoarele semnificații:

- `function` – cuvânt cheie care declară fișierul ca fișier funcție.
- `nume` – numele funcției; reprezintă numele sub care se salvează fișierul .m (extensia .m nu face parte din nume); acest nume nu poate fi identic cu cel al unui fișier .m deja existent.
- `parametrii_ieșire` – reprezintă parametrii de ieșire ai funcției; trebuie separați prin virgulă și cuprinși între paranteze drepte.
- `parametrii_intrare` – reprezintă parametrii de intrare ai funcției; trebuie separați prin virgulă și cuprinși între paranteze rotunde.

***Atenție:** Sintaxa de mai sus se simplifică în cazul în care funcția are doar parametrii de intrare sau doar parametrii de ieșire (dispar parantezele drepte, respectiv parantezele rotunde).*

### 1.2.9. Operații aritmetice. Operații asupra numerelor complexe

#### • *Operatori aritmetici*

- `+` - adunare
- `-` - scădere
- `*` - înmulțire
- `.*` - înmulțire între două matrice (sau vectori) element cu element
- `/` - împărțire
- `./` - împărțire între două matrice (sau vectori) element cu element
- `^` - ridicare la putere
- `.^` - ridicare la putere a unei matrice (sau vector) element cu element
- `\` - transpunere și conjugare
- `.\` - transpunere

Să se definească următoarele elemente:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}, B = \begin{pmatrix} 1-i & 2+i & 0 \\ -i & 3 & i \end{pmatrix}, a = (1+i \ 3 \ 0), b = \begin{pmatrix} 1/3 \\ 2^2 \end{pmatrix}$$

## 1. INTRODUCERE ÎN MATLAB

---

Verificați și explicați următoarele exemple:

$A+B$  ,  $A+3$  ,  $B-2$  ,  $B-i$  ,  $a+a$  ,  $A*B$  ,  $A.*B$  ,  $A*B.'$  ,  $a.*a$  ,  $a*a$  ,  $A*a$  ,  $A*a'$  ,  $A*a.'$  ,  $A'*b$  ,  $B'$  ,  $B.'$  ,  $A'$  ,  $A.'$  ,  $A^2$  ,  $A.^2$  ,  $1-b$  ,  $b'*b$  ,  $b*b$  ,  $b.*b$  ,  $b^3$  ,  $b.^3$  ,  $2/A$  ,  $2./A$  ,  $2/b$  ,  $2./b$  .

- **Operații asupra numerelor complexe**

- `abs` – calculează valoarea absolută (modulul).
- `angle` – calculează faza.
- `conj` – calculează complex conjugatul.
- `real` – extrage partea reală.
- `imag` – extrage partea imaginară.

Verificați comenzile utilizând drept argumente elementele  $A$ ,  $B$ ,  $a$  și  $b$  definite anterior.

### **E2. Exercițiu:**

Folosind sintaxele și indicațiile din secțiunile 1.2.8. și 1.2.9. realizați o funcție MATLAB care având drept parametri de intrare un vector linie cu elemente complexe returnează ca parametri de ieșire:

- media aritmetică a părților reale ale elementelor vectorului;
- un vector ce conține elementele vectorului inițial ridicate la pătrat;
- o matrice obținută din înmulțirea vectorului inițial cu transpusul său.

### 1.2.10. Funcții matematice uzuale

- **Funcțiile radical, exponențială și logaritm**

- `sqrt` – extragere radical de ordinul 2 (rădăcina pătrată).
- `exp` – calculează exponențiala (puteri ale numărului  $e$ ).
- `log` – calculează logaritmul natural (logaritm în baza  $e$ ).
- `log2` – calculează logaritmul în bază 2.
- `log10` – calculează logaritmul zecimal (logaritm în baza 10).
- `pow2` – calculează puteri ale lui 2.

- **Funcțiile trigonometrice directe**

- `sin` – calculează sinusul
- `cos` – calculează cosinusul
- `tan` – calculează tangenta
- `cot` – calculează cotangenta
- `sec` – calculează secanta
- `csc` – calculează cosecanta



## 1. INTRODUCERE ÎN MATLAB

---

- **Funcțiile trigonometrice inverse**
  - `asin` – calculează arcsinus
  - `acos` – calculează arccosinus
  - `atan` – calculează arctangenta
  - `atan2` – calculează arctangenta dacă argumentul este complex
  - `acot` – calculează arccotangenta
  - `asec` – calculează arcsecanta
  - `acsc` – calculează arccosecanta
- **Funcțiile hiperbolice directe**
  - `sinh` – calculează sinusul hiperbolic
  - `cosh` – calculează cosinusul hiperbolic
  - `tanh` – calculează tangeta hiperbolică
  - `coth` – calculează cotangenta hiperbolică
  - `sech` – calculează secanta hiperbolică
  - `csch` – calculează cosecanta hiperbolică
- **Funcțiile hiperbolice inverse**
  - `asinh` – calculează arcsinus hiperbolic
  - `acosh` – calculează arccosinus hiperbolic
  - `atanh` – calculează arctangenta hiperbolică
  - `acoth` – calculează arcotangenta hiperbolică
  - `asech` – calculează arcsecanta hiperbolică
  - `acsch` – calculează arccosecanta hiperbolică

Pentru informații despre modul de utilizare al acestor funcții folosiți comanda `help` însoțită de numele funcției dorite (vezi sintaxa de la pagina 2).

### 1.2.11. Funcții destinate analizei de date

- **`sum, prod` – Suma și produsul**

*Sintaxe:*

Dacă `v` este un vector și `M` este o matrice atunci

- `sum(v)` – calculează suma elementelor vectorului `v`.
- `prod(v)` – calculează produsul elementelor vectorului `v`.
- `sum(M)` – returnează un vector linie având ca elemente suma elementelor fiecărei coloane din matricei `M`.
- `prod(M)` – returnează un vector linie având ca elemente produsul elementelor fiecărei coloane din matricei `M`.

## 1. INTRODUCERE ÎN MATLAB

---

Se vor defini următoarele elemente:  $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ ,  $a = (1 \ 3 \ 8)$

Să se verifice următoarele *exemple*:

`sum(a)`

`sum(A)`

`sum(sum(A))`

`prod(a)`

`prod(A)`

`prod(prod(A))`

- **max,min** – *Maximul și minimul*

*Sintaxe:*

Dacă  $v$  este un vector și  $M$  este o matrice atunci

- `max(v)` – returnează elementul maxim al vectorului  $v$ .
- `min(v)` – returnează elementul minim al vectorului  $v$ .
- `[m,p]=max(v)` – returnează elementul maxim al vectorului ( $m$ ) precum și indicele elementului maxim ( $p$ ); dacă există maxime multiple se returnează indicele primului dintre ele.
- `[m,p]=min(v)` – returnează elementul minim al vectorului ( $m$ ) precum și indicele elementului minim ( $p$ ); dacă există minime multiple se returnează indicele primului dintre ele.
- `max(M)` – returnează un vector linie având ca elemente maximul elementelor din fiecare coloană a matricei  $M$ .
- `min(M)` – returnează un vector linie având ca elemente minimul elementelor din fiecare coloană a matricei  $M$ .
- `[m,p]=max(M)` – returnează un vector linie  $m$  având ca elemente maximul elementelor din fiecare coloană a matricei  $M$  precum și un vector linie  $p$  ce conține poziția maximului respectiv în cadrul fiecărei coloane.
- `[m,p]=min(M)` – returnează un vector linie  $m$  având ca elemente minimul elementelor din fiecare coloană a matricei  $M$  precum și un vector linie  $p$  ce conține poziția minimului respectiv în cadrul fiecărei coloane.

*Verificați* sintaxele anterioare folosind matricea  $A$  și vectorul  $a$  definite mai sus.

- **mean** – *Media aritmetică*

*Sintaxe:*

Dacă  $v$  este un vector și  $M$  este o matrice atunci

- `mean(v)` – calculează media aritmetică a elementelor vectorului  $v$ .
- `mean(M)` – returnează un vector linie având ca elemente media aritmetică a elementelor fiecărei coloane din matricei  $M$ .

*Verificați* sintaxele anterioare folosind matricea  $A$  și vectorul  $a$  definite mai sus.

**E3.** *Reluați exercițiul de la pagina 14 folosind funcția mean.*

## 1. INTRODUCERE ÎN MATLAB

---

### 1.2.12. Reprezentări grafice

- **plot, stem** – *Reprezentări grafice în coordonate liniare*

*Sintaxe:*

Dacă  $v$  este un vector și  $M$  este o matrice atunci

- `plot(v)` – dacă  $v$  este un vector cu *elemente reale* se vor reprezenta grafic elementele sale în funcție de indici (primul indice este 1, ultimul indice este egal cu lungimea vectorului); dacă  $v$  este un vector cu *elemente complexe* atunci reprezentarea sa se va face în funcție de partea reală (pe abscisă) și de partea imaginară (pe ordonată) a elementelor sale.
- `plot(M)` – se vor reprezenta pe același grafic coloanele matricei  $M$  (fiecare din coloanele matricei este privită ca un vector și reprezentat ca în sintaxa precedentă);

Dacă  $x$  și  $y$  sunt doi vectori de *aceeași lungime* și  $N$  este o matrice de *aceeași dimensiune* cu matricea  $M$  atunci

- `plot(x, y)` – se vor reprezenta grafic elementele vectorului  $y$  în funcție de elementele vectorului  $x$ ; *dacă lungimea vectorului  $x$  nu este egală cu lungimea vectorului  $y$  reprezentarea nu este posibilă.*
- `plot(x, M)` – dacă lungimea vectorului  $x$  este egal cu numărul de linii al matricei  $M$  atunci se vor reprezenta pe același grafic coloanele matricei  $M$  în funcție de elementele vectorului  $x$ ; dacă lungimea vectorului  $x$  este egal cu numărul de coloane al matricei  $M$  atunci se vor reprezenta pe același grafic liniile matricei  $M$  în funcție de elementele vectorului  $x$ ; *dacă lungimea vectorului  $x$  nu este egală cu una din dimensiunile matricei  $M$  atunci reprezentarea nu este posibilă.*
- `plot(M, N)` – se vor reprezenta pe același grafic coloanele matricei  $N$  în funcție de coloanele matricei  $M$  (coloana  $k$  din matricea  $N$  va fi reprezentată în funcție de coloana  $k$  din matricea  $M$ , unde  $k = 1, 2, \dots$ , numărul de coloane); *dacă cele două matrice nu au aceeași dimensiune atunci reprezentarea nu este posibilă.*

Pentru reprezentarea mai multor grafice în aceeași fereastră grafică, utilizând o singură comandă, folosim

- `plot(x1, y1, x2, y2, ..., xn, yn)` - se vor reprezenta pe același grafic  $y_1$  în funcție de  $x_1$ ,  $y_2$  în funcție de  $x_2$ , ...,  $y_n$  în funcție de  $x_n$  (pot fi vectori sau matrice); rămân valabile considerentele făcute în sintaxele anterioare, referitor la cazurile când avem vectori sau matrice; *dacă  $x_1, y_1, x_2, y_2, \dots, x_n, y_n$  nu au aceeași dimensiune atunci reprezentarea nu este posibilă.*

**Atenție:**

*Se pot folosi diverse linii, markere și culori de reprezentare a graficelor. Vezi `help plot`.*

## 1. INTRODUCERE ÎN MATLAB

---

Funcția `stem` realizează o reprezentare în formă “discretă” a datelor. Pentru variantele de MATLAB 5 se pot folosi oricare din sintaxele prezentate la `plot`, exceptând ultima sintaxă. Pentru variantele de MATLAB 4 argumentele de intrare ale funcției `stem` nu pot fi decât vectori, iar reprezentarea grafică nu se poate face decât cu o singură culoare (se pot folosi însă mai multe tipuri de linii și markere).

Vezi `help stem`.

- **`loglog`, `semilogx`, `semilogy`** – *Reprezentări grafice în coordonate logaritmice*

Pentru acest tip de reprezentări se folosesc funcțiile `loglog`, `semilogx` și `semilogy`. Sintaxele rămân aceleași ca la funcția `plot`, singura deosebire fiind modul de scalare al axelor. Astfel funcția `loglog` scalează ambele axe (abscisa și ordonata) folosind logaritmul în bază 10, deci pe axe vom avea puteri ale lui 10. Funcția `semilogx` realizează același tip de scalare însă numai pe abscisă iar funcția `semilogy` procedează în același mod însă numai pe ordonată.

- **`subplot`** – *Divizarea ferestrei grafice*

Dacă dorim ca fereastra grafică să conțină mai multe reprezentări grafice se poate folosi funcția `subplot` care împarte fereastra grafică în mai multe “miniferestre”, în fiecare dintre acestea putând fi plasat câte un grafic. Fereastra grafică este astfel privită sub forma unei matrice cu  $m$  linii și  $n$  coloane, deci în total  $m \cdot n$  “miniferestre”. Numărarea acestor “miniferestre” se face pe linii. De exemplu, dacă vrem să împărțim fereastra grafică în  $3 \cdot 3 = 9$  “miniferestre” vom avea următoarea ordine:

1	2	3
4	5	6
7	8	9

*Sintaxa:*

- `subplot(m, n, p)` – împarte fereastra grafică într-o matrice  $m \times n$  ( $m \cdot n$  “miniferestre”), iar  $p$  reprezintă numărul fiecărei “miniferestre” în matricea grafică respectivă (numărarea se face pe linii); sintaxa respectivă este urmată de comanda propriu-zisă de afișare a graficului, care poate fi oricare din cele prezentate până acum (`stem`, `plot`, `loglog`, `semilogx`, `semilogy`).

- **`axis`** – *Schimbarea limitelor axelor*

Dacă se dorește vizualizarea numai a unei anumite porțiuni dintr-un grafic, corespunzătoare unor anumite intervale pe abscisă și ordonată, se va folosi comanda `axis`.

## 1. INTRODUCERE ÎN MATLAB

---

### *Sintaxa:*

- `axis([x0 x1 y0 y1])` – pe abscisa se va vizualiza între valorile `x0` și `x1` iar pe ordonată între `y0` și `y1`; această sintaxă se plasează după comanda de reprezentare grafică.

- **`title`, `xlabel`, `ylabel`, `gtext`, `grid`** – *Precizarea titlului graficului și a etichetelor axelor. Trasarea unei rețele pe grafic. Plasarea unui text pe grafic.*

### *Sintaxe:*

- `title('text')` – plasează deasupra graficului, ca titlu, textul `text`.
- `xlabel('text')` – textul `text` devine eticheta de pe abscisă.
- `ylabel('text')` – textul `text` devine eticheta de pe ordonată.
- `grid` – trasează pe grafic o rețea de linii, înlesnind astfel citirea graficului.
- `gtext('text')` – plasează pe grafic textul `text` (folosind mouse-ul).

Toate aceste sintaxe urmează după comanda de reprezentare grafică.

- **`hold`** – *Suprapunerea succesivă a graficelor*

Dacă dorim să reținem graficul curent și să adăugăm în aceeași fereastră grafică următoarele reprezentări grafice se poate folosi funcția `hold`.

### *Sintaxa:*

- `hold on` – reține graficul curent și adaugă în aceeași fereastră grafică următoarele reprezentări grafice.
- `hold off` – dacă se dorește în continuare reprezentarea în ferestre grafice separate (dezactivează comanda `hold on`).

### **Atenție:**

*Dacă se dorește în cadrul unui program reprezentarea mai multor grafice în ferestre separate, fiecare comandă grafică va trebui să fie precedată de un nume de forma `figure(n)`, unde `n` este numărul figurii respective. În caz contrar, la sfârșitul execuției programului va apărea numai ultima reprezentare grafică (se va folosi o singură fereastră grafică ce va fi “ștearsă” de fiecare dată la întâlnirea unei noi comenzi grafice).*

**Exemple:** Se vor tasta direct în fereastra de comenzi, *explicând rezultatele:*

```
x=0:0.2:2*pi;
size(x)
s=sin(x);
size(s)
plot(s),grid,title('sinus'),xlabel('n')
stem(s),grid,title('sinus'),xlabel('n')
n=linspace(min(x),max(x),length(x));
size(n)
```

## 1. INTRODUCERE ÎN MATLAB

---

```
plot(n,s,'r*'),grid,title('sinus'),xlabel('n')
hold on
stem(n,s),grid,xlabel('n'),ylabel('amplitudine')
hold off
figure(1)
plot(n,s),grid,axis([0 pi min(s) max(s)])
figure(2)
plot(n,s,n,s-pi/2),grid
// se închid ambele ferestre grafice și se continuă tastarea în fereastra de comenzi
c=cos(x);
subplot(2,1,1),stem(n,s),title('sinus'),grid
subplot(2,1,2),stem(n,c),title('cosinus'),grid
M=[c;s];
plot(n,M),grid
N=[n;n];
plot(N,M),grid
plot(N',M'),grid
z=1:1000;
p=z.^2;
plot(z,p),grid
loglog(z,p),grid
semilogx(z,p),grid
semilogy(z,p),grid
gtext('Ultimul grafic!!!')
// cu ajutorul mouse-ului plasați pe grafic textul respectiv, în poziția dorită.
```

### **E4. Exercițiu:**

Realizați un program MATLAB care generează două secvențe  $x_1[n]$  și  $x_2[n]$  definite astfel:

$$x_1[n] = \sin\left(\frac{\pi}{17}n\right), \quad \text{pentru } -15 \leq n \leq 25$$

$$x_2[n] = \cos\left(\frac{\pi}{\sqrt{23}}n\right), \quad \text{pentru } 0 \leq n \leq 50$$

și reprezintă cele două secvențe în două moduri:

- figura 1 – în același sistem de coordonate (pe același grafic);
- figura 2 – folosind două “miniferestre” grafice plasate una sub alta.

Reprezentați cele două figuri folosind comanda `plot` iar apoi încercați reprezentarea lor folosind comanda `stem`.