# Teoria sistemelor. Laborator 1: Introducere în Matlab

*Obiectiv:*

- Introducere în MATLAB
- Recapitularea unor noţiuni de matematică necesare pentru cursul de Teoria sistemelor: polinoame, transformata Laplace, numere complexe

## 1 Introducere

Matlab este un mediu software pentru calcule ştiintifice şi inginereşti, dezvoltat de Mathworks Inc. (www.mathworks.com). Numele vine de la "**Mat**rix **Lab**oratory" deoarece limbajul vizează soluţiile problemelor cu vectori şi matrici. În acest capitol este prezentată o scurtă introducere a noţiunilor de bază din Matlab, utile pentru rezolvarea problemelor propuse în îndrumator.

## 2 Matlab

La pornire, fereastra principală din Matlab arată ca în Figura 1.

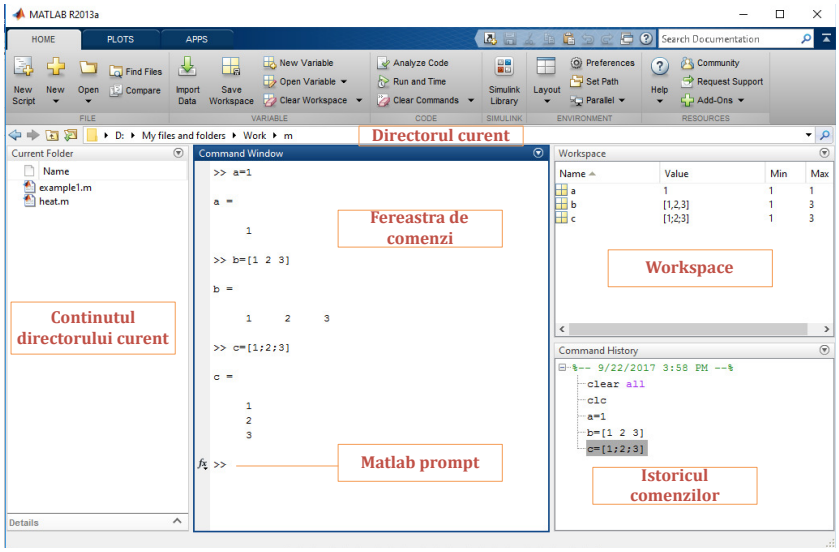


Figura 1: Fereastra principală

Ferestrele din Matlab sunt următoarele:

- Command window - pentru a introduce comenzi din linia de comandă indicată prin (>>)
- Directorul curent - pentru accesarea şi manipularea fişierelor
- Workspace (spaţiul de lucru)- vizualizarea datelor create în sesiunea de lucru curentă
- Command history - vizualizarea istoricului comenzilor

Pentru a ieşi din Matlab, scrieţi *quit* sau *exit* în linia de comandă.

Pentru a întrerupe execuţia unei comenzi: apasaţi Ctrl+C.

# 3 Variabile şi instructiuni

Instrucţiunile sunt de forma:

```
>> variabla = expresie
```

Semnul egal implică atribuirea expresiei variabilei. În exemplul de mai jos, variabila *a* ia valoarea 1 dupa apăsarea tastei **Enter**.

```
>> a = 1
a =
     1
```

Valoarea variabilei este afişată după execuţia instrucţiunii. Dacă instrucţiunea este urmată de punct şi virgulă, acest lucru nu se mai afişează.

```
>> b = 2;
```

Valorile variabilelor *a* şi *b* pot fi văzute fie în *Workspace*, fie prin scrierea numelui variabilei în linia de comandă, fără punct şi virgulă.

Matlab este sensibil la litere mari şi mici (*case senzitive*). Numele unei variabile începe cu o literă şi poate fi urmată de alte litere, cifre sau caracterul *underscore* _.

Matlab poate fi utilizat în *modul calculator*. În cazul în care numele variabilei şi semnul egal lipsesc, atunci rezultatul este stocat într-o variabilă numita *ans* (de la *answer*). Această variabilă este suprascrisă la fiecare nou calcul.

```
>> 2.5*3
ans =
   7.5000
>> 10-2
ans =
    8
```

# 4 Vectori şi matrici

Variabilele din Matlab sunt *matrici* multidimensionale, indiferent de tipul de date, [2]. Un *vector linie* se creează prin introducerea elementelor între paranteze pătrate, separate prin spatiu sau virgulă.

```
>> a = [1 2 3 4 5 6]

   a =
       1  2  3  4  5  6
```

Un *vector coloană* (un vector cu mai multe linii şi o coloană), poate fi creat prin introducerea elementelor între paranteze pătrate, separate prin punct şi virgula (;).

```
>> b = [1;2;3;4]

   b =
       1
       2
       3
       4
```

Pentru crearea unei matrici, se separă elementele de pe aceeaşi linie prin spatiu sau prin virgulă şi liniile prin punct şi virgulă.

```
>> A = [1 2 3 4;5 6 7 8;9 10 11 12]

   A =
          1    2    3     4
          5    6    7     8
          9   10   11    12
```

Crearea unui vector linie cu valori cuprinse între *valoarea_initiala* şi *valoarea_finala*, folosind un increment *inc*, se obţine astfel:

```
>> vector = valoare_initiala:inc:valoare_finala
```

Dacă incrementul lipseşte, acesta se consideră implicit egal cu 1. De exemplu:

```
>> x=1:5
x =
     1     2     3     4     5

>> y=0:0.2:1
y =
         0    0.2000    0.4000    0.6000    0.8000    1.0000

>> z=10:-1:5
z =
    10     9     8     7     6     5
```

Se poate face referire la elementele dintr-o matrice, prin specificarea liniei şi a coloanei, între paranteze rotunde:

```
>> A(linie,coloana)
```

Elementul de pe a doua linie şi a treia coloană din matricea $A$ creată anterior, se poate afişa astfel:

```
>> A(2,3)

ans =
     7
```

Două puncte (:), folosite în loc de numarul liniei sau al coloanei, face referire la toate liniile sau coloanele din matrice. Toate elementele de pe a treia coloana din matricea $A$ se obţin astfel:

```
>> A(:,3)

ans =
     3
     7
    11
```

Concatenarea mai multor vectori se face între paranteze pătrate. Vectorii $x$ şi $y$ creaţi anterior pot fi concatenaţi astfel:

```
>> [x y]
ans =
  Columns 1 through 8
    1.0000   2.0000   3.0000   4.0000   5.0000   0   0.2000   0.4000

  Columns 9 through 11
```

```
    0.6000    0.8000    1.0000
```

O noua valoare este ataşată dupa a treia coloană din matricea $A$ şi rezultatul este stocat în noua variabilă $Am$:

```
>> Am = [A(:,3); 22]

Am =
     3
     7
    11
    22
```

# 5 Operatori

Operatorii matematici uzuali pot fi folosiţi în expresii iar ordinea operatorilor aritmetici poate fi modificată folosind paranteze rotunde. Cei mai folosiţi operatori se găsesc în Tabelul 1.

| **Simbol** | **Rol** | **Simbol** | **Rol** |
|---|---|---|---|
| $+$ | Adunare | $.*$ | Înmulţirea element cu element |
| $-$ | Scădere | $./$ | Împărţirea element cu element |
| $*$ | Înmulţire | $.\hat{}$ | Ridicarea la putere element cu element |
| $/$ | Împărţirea | $'$ | Transpusa |
| $\hat{}$ | Ridicarea la putere | $\backslash$ | Împarţirea la stânga |

Tabelul 1: Operatori matematici

Pentru a aduna 2 la fiecare element din vectorul $a$, se procedeaza astfel:

```
>> a=[1 2 3 4 5 6];
>> b = a + 2
     b =
          3  4  5  6  7  8
```

Pentru a aduna doi vectori, aceştia trebuie să aibă aceeaşi lungime. Vectorul $c$ reprezintă suma vectorilor $a$ şi $b$ creati anterior:

```
>> c = a + b
     c =
          4  6  8  10  12  14
```

La fel se procedează şi pentru a aduna sau a scădea elementele matricilor cu aceeaşi dimensiune.

```
>> A=[1 0;0 1], B=[0 1; 1 0], C=[1 2; 3 4], D=A+B-C
A =
     1     0
     0     1
B =
     0     1
     1     0
C =
     1     2
     3     4
D =
     0    -1
    -2    -3
```

Matricile $B$ şi $C$ se pot înmulţi. Atenţie! Înmulţirea matricilor este posibilă dacă numărul de coloane ale primei matrici este egal cu numărul liniilor celei de a doua.

```
>> B*C
ans =
     3     4
     1     2
>> C*B
ans =
     2     1
     4     3
```

O matrice pătrată, de exemplu $C$, poate fi înmulţită cu ea însaşi prin ridicarea la putere.

```
>> C^3
     ans =
         37    54
         81   118
```

Pentru a înmulţi element cu element .* două matrici, acestea trebuie să aibă aceeaşi dimensiune. Folosind operatorul ./, se realizează împarţirea element cu element.

```
>> B.*C
ans =
     0     2
     3     0
>> B./C
ans =
         0    0.5000
    0.3333         0
```

Folosind operatorul .^ se realizează ridicarea la o putere a fiecărui element din matrice.

```
>> C.^3
ans =
     1     8
    27    64
```

Transpusa unei matrici se obtine folosind apostrof:

```
>> E = C'
E =
     1     3
     2     4
```

Pentru a afla soluţia ecuaţiei $\mathbf{Ax} = \mathbf{b}$, dacă aceasta există, se foloseşte împartirea la stânga (\). De exemplu, se consideră sistemul de ecuaţii liniare:

$$\begin{cases} 4x_1 + 3x_2 &= 5 \\ 2x_2 + x_2 &= 6 \end{cases}$$

Acesta poate fi scrisă sub formă matricială astfel:

$$\begin{bmatrix} 4 & 3 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 6 \end{bmatrix}, \text{ where } A = \begin{bmatrix} 4 & 3 \\ 2 & 1 \end{bmatrix}, \; x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \; b = \begin{bmatrix} 5 \\ 6 \end{bmatrix}$$

Soluţia sistemului liniar $\mathbf{Ax} = \mathbf{b}$ se obţine, în Matlab, astfel:

```
>> A=[4 3; 2 1]; b=[5;6]; x=A\b
x =
    6.5000
   -7.0000
```

# 6 Numere complexe

În Matlab se poate opera cu numere complexe, cu parte reală şi parte imaginară. Unitatea imaginară, al cărei patrat este $-1$, este stocată în variabila $i$ sau $j$. Aceste doua variabile pot fi suprascrise! Numărul complex $x = 2 + 3i$ şi conjugatul său $y = 2 - 3i$ pot fi declarate astfel, folosind fie $i$, fie $j$.

```
>> x=2+3i, y=2-3j
x =
   2.0000 + 3.0000i
y =
   2.0000 - 3.0000i
```

# 7 Funcţii predefinite

În Matlab sunt incluse funcţii predefinite pentru multe operaţii matematice. Codul sursă al acestora nu este, de obicei, disponibil utilizatorului. Pentru a determina modul în care se foloseşte o anumită funcţie, folosiţi sintaxa use *help nume_functie* sau *doc nume_functie*. *Help* afişează informaţiile în linia de comandă, iar *doc* deschide o fereastră nouă în care se regăseşte descrierea funcţiei, exemple de utilizare şi alte informaţii.

```
>> help det
>> doc inv
```

Câteva funcţii trigonometrice uzuale sunt prezentate în Tabelul 2. Atât paramtrii de intrare, cât şi rezultatul funcţiei, se exprima în radiani.

Tabelul 2: Funcţii trigonometrice

| | |
|---|---|
| sin(x) | Sinusul elementelor din x (în radiani) |
| cos(x) | Cosinusul elementelor din vectorul x (în radiani) |
| asin(x) | Arcisnus pentru elementele din x (rezultatul în radiani) |
| acos(x) | Arccosinus pentru elementele din x (rezultatul în radiani) |
| tan(x) | Tangenta elementelor din vectorul x (în radiani) |
| atan(x) | Arctangenta elementelor din x (rezultatul în radiani) |

În exemplul de mai jos, se calculează sinusul unghiurilor $90^o$, $60^o$ şi $30^o$, unde *pi* este o constantă predefinită în Matlab.

```
>> angles_degrees=[90 60 30]
angles_degrees =
    90    60    30
>> sin(angles_degrees*pi/180)
ans =
    1.0000    0.8660    0.5000
```

În Tabelul 3 este dată o listă cu cele mai utilizate funcţii matematice.

Partea reală, partea imaginară şi modulul unui număr complex se calculează astfel:

```
>> x=4+5j;
>> re=real(x), im=imag(x), ab=abs(x)
re =
     4
```

```
im =
     5
ab =
    6.4031
```

Tabelul 3: Funcţii matematice de bază

| | |
|---|---|
| abs(x) | Valoarea absoluta a elementelor din x |
| sqrt(x) | Radical din x |
| imag(x) | Partea imaginară a lui x |
| real(x) | Partea reală a lui x |
| conj(x) | Conjugatul lui x |
| log(x) | Logaritm natural |
| log10(x) | Logaritm în baza 10 |
| exp(x) | Exponentiala |

În Tabelul 4 sunt date câteva funcţii folosite pentru proprietăţile unei matrici sau pentru construirea matricilor.

Tabelul 4: Proprietăţile unei matrici şi operatii pe matrici

| | |
|---|---|
| inv(A) | Inversa matricei A |
| eig(A) | Valorile proprii ale matricei A |
| det(A) | Determinantul matricei A |
| rank(A) | Rangul |
| eye | Crează o matrice unitate |
| ones, zeros | Creează o matrice compusă din unu sau zero |
| diag | Creează o matrice diagonală |

Pentru o matrice pătrată $A$, inversa şi determinantul se calculează astfel:

```
>> A=[1 2;3 4];
>> inv(A)
ans =
   -2.0000    1.0000
    1.5000   -0.5000
>> det(A)
ans =
    -2
```

Matricea unitate de dimensiune 3 şi un vector cu toate elementele 1, de dimensiune 1-pe-3, se creează astfel:

```
>> B=eye(3)
B =
     1     0     0
     0     1     0
     0     0     1
>> C=ones(1,3)
C =
     1     1     1
```

# 8 Polinoame

In Matlab, polinoamele sunt reprezentate printr-un vector linie care conţine coeficienţii ordonaţi în ordinea descrescătoare a puterilor variabilei. Astfel, pentru un polinom de ordinul $n$, vom declara un vector de

lungime $n+1$. Polinoamele:

$$p(s) = s^4 + 2s^3 - 3s^2 + 4s - 5 \text{ si } q(s) = s^3 + 6$$

sunt declarate ca vectori astfel:

```
>> p = [1 2 -3 4 -5];
>> q = [1 0 0 6];
```

Câteva funcţii folosite pentru polinoame sunt date în Tabelul 5. De exemplu, rădăcinile polinomului $p(s)$,

Tabelul 5: Functii pentru polinoame

| | |
|---|---|
| roots(p) | Rădăcinile polinomului p |
| polyval(p,v) | Polinomul p, evaluat în v |
| conv(p,q) | Înmulţirea polinoamelor |
| deconv(p,q) | Împărţirea polinoamelor |

se obţin:

```
>> roots(p)
ans =
  -3.3719 + 0.0000i
   1.1103 + 0.0000i
   0.1308 + 1.1482i
   0.1308 - 1.1482i
```

Pentru înmulţirea polinoamelor $p(s)$ şi $q(s)$, se foloseşte convoluţia. Pentru a împarţi polinomul $p(s)$ la $q(s)$ se foloseşte funcţia *deconv*. Aceasta returnează câtul şi restul, sub forma a doi vectori (două polinoame). Numele variabilelor corespunzătoare acestor doua polinomane trebuie sa fie incluse intre paranteze patrate la apelarea functiei. Pentru a determina modul în care se foloseşte această funcţie, scrieţi *help deconv* în linia de comandă.

```
>> pq=conv(p,q)
pq =
     1     2    -3    13    13   -27    36   -45
>> [qu,re]=deconv(p,q)
qu =
     1     2
re =
     0     0    -3    -5   -23
```

# 9 Reprezentarea grafică

Matlab-ul are posibilităţi variate pentru reprezentările grafice, de la grafice 2D, la grafice tri-dimensionale.

Funcţia *plot* poate avea diferite forme, în funcţie de parametrii de intrare.

Pentru a reprezenta vectorul $y$ în funcţie de vectorul $x$, se foloseşte *plot(x,y)*. Pentru a reprezenta un semnal sinusoidal în funcţie de timp, trebuie declarat un vector de timp pentru care se aplică functia *sin*. În exempul de mai jos, vectorul de timp este format de la 0 la $4\pi$, cu pasul 0.1.

```
>> x=0:0.1:4*pi;
>> y=sin(x);
>> plot(x,y)
```

Pentru mai multe seturi de date, se foloseste forma generală *plot(x1,y1, x2,y2, ...)*. Reprezentarea pe acelaşi grafic a unui semnal sinusoidal şi cosinus se realizează astfel, într-o figura nouă la care se adaugă grid:

Tabelul 6: Reprezentări grafice

| | |
|---|---|
| plot(x,y) | Reprezentarea grafică a vectorului y în funcţie de vectorul x |
| mesh, surf | Crează o suprafaţă 3D |
| figure | Crează o fereastră nouă pentru figuri |
| grid, grid on/off | Grid on/off |
| title('text') | Adaugă text deasupra figurii |
| xlabel('text') | Denumeşte axa x cu 'text' |
| ylabel('text') | Denumeşte axa y cu 'text' |
| legend(string) | Afişează legenda |
| hold, hold on/off | Comută între starea 'hold' on şi off/ |
| subplot | Creează mai multe reprezentări în aceeaşi figură |

```
>> x=0:0.1:4*pi;
>> figure, plot(x,sin(x),x,cos(x))
>> grid on
```

În cazul în care o figură este deschisă, o comandă 'plot' nouă va şterge fereastra şi va suprascrie în aceeaşi fereastră noul grafic. Functia *figure* deschide o nouă fereastra în care reprezintă noul grafic.

Culorile graficelor sunt alese automat. Utilizatorul poate să schimbe stilul şi culoarea graficului (curbei) prin folosirea formei generale *plot(x1,y1,'options1', x2,y2, 'options2', ...)*. Toate opţiunile posibile se pot vedea folosind comanda *help plot*.

În exemplul urmator, cele două grafice reprezentate mai sus sunt trasate folosind $*$ de culoare roşie, pentru primul set de date, iar pentru al doilea set de date se foloseşte linia punctată de culoare neagră. În aceeaşi figură se afişează şi legenda.

```
>> x=0:0.1:4*pi;
>> figure, plot(x,sin(x),'r*',x,cos(x),'k:')
>> grid on
>> legend('sine', 'cosine')
```

În exemplul urmator, sunt reprezentate în funcţie de $x$ între $[-1, 1]$, funcţiile $x^2$ şi $x^3$. Se adaugă un titlu figurii şi etichete pentru axe.

```
>> x=-1:0.1:1;
>> figure, plot(x, x.^2, x, x.^3)
>> grid on
>> xlabel('x'), ylabel('x^2, x^3'), title('x p\u{a}trat si x cub')
```

În cazul în care se folosea operaţia simplă de multiplicare, se încerca înmulţirea vectorului $x$ cu el însuşi, rezultând o eroare. În acest caz, a fost nevoie de pătratul (sau cubul) fiecarui element din vectorul $x$, astfel s-a folosit operatorul .^ .

Daca se doreşte adăugarea unui nou grafic la unul deja existent, se foloseşte comanda *hold on*. Starea aceasta poate fi resetată folosind *hold off*.

```
>> x=-5:0.2:5;
>> figure, plot(x,sin(x), 'r'), grid on
>> hold on
>> plot(x,cos(x), 'm')
>> hold off
```

Comanda *subplot* este folosită pentru crearea uneia sau mai multor sub-figuri în aceeaăi fereastră. *Subplot(m,n,p)* sau *subplot(mnp)* va împarţi figura curentă într-o matrice m-pe-n şi va seta graficul curent pe poziţia p. În exemplul următor, se reprezintă funcţia sinus pentru mai multe frecvenţe, iar graficele se reprezintă într-o figură împarţită pe trei rânduri şi o coloană.

```
>> x=-2:0.1:2;
>> subplot(311), plot(x,sin(2*x)), xlabel('x'), ylabel('sin(2x)')
>> subplot(312), plot(x,sin(3*x)), xlabel('x'), ylabel('sin(3x)')
>> subplot(313), plot(x,sin(4*x)), xlabel('x'), ylabel('sin(4x)')
```

# 10 Fişiere m

Fişierele *m* sunt fişiere text în care se scriu comenzi. Acestea se creează folosind editorul din Matlab şi se salvează cu extensia *.m*. Fişierele *m* sunt de doua tipuri:

- Fişiere script - conţin instrucţiuni ca cele introduse în linia de comandă, care se execută secvenţial. Toate variabilele folosite în aceste fisiere sunt vizibile în workspace.
- Funcţii definite de utilizator - sunt fişiere-program care au parametri de intrare şi de iesire. Variabilele folosite în interiorul funcţiei sunt stocate doar local şi nu se pot vedea în workspace.

Înainte de a edita şi a rula fişiere script noi sau de a apela funcţii, setaţi directorul curent ca directorul în care lucraţi. Acest lucru se face în interfaţa principală din Matlab.

Important! Numele fişierelor nu trebuie să coincidă cu nume de funcţii predefinite în Matlab. De exemplu, nu este indicat saă salvaţi un fisier cu numele *plot.m*, *sin.m* sau *roots.m*.

## 10.1 Fisiere script

Pentru a crea si a rula un script nou:

- Deschideţi editorul Matlab sau selectaţi *New script* în fereastra principală.
- Scrieţi comenzile aşa cum ar fi ele introduse în linia de comandă.
- Salvaţi fişierul în directorul de lucru curent.
- Rulaţi scriptul în oricare din următoarele 3 moduri:
  - Apasaţi *F5* în editorul Matlab
  - Apasaţi butonul verde *Save and run* din editorul Matlab
  - Scrieţi numele fişierului, fără extensie, în linia de comandă.

Este recomandat să începeţi un fişier script cu trei comenzi: *close all*, *clear all* şi *clc*. Comanda *close all* închide toate figurile deschise, comanda *clear all* şterge toate variabilele memorate în workspace şi *clc* curaţă fereastra de comenzi.

Comentarile se adaugă în program folosind simblolul (%),

Un exemplu de program este ilustrat mai jos:

Listing 1: script_example.m

```
close all
clear all
clc
%-------------------------------------------------------
% afiseaza trei functii sinusoidale cu faza diferita
% pe un interval de timp intre 1 si 15
%-------------------------------------------------------
t=0:0.01:15;    % se introduce vectorul timp
s1=sin(t);      % se creaza primul sinus
s2=sin(t-1);    % se creaza al doilea sinus deplasat
s3=sin(t-2);    % se creaza al treilea sinus deplasat mai mult

% se plaseaza toate cele trei grafice in aceeasi fereastra
% se adauga grid, etichete pe axe si titlu
plot(t,s1,t,s2,t,s3, 'LineWidth',2), grid on
xlabel('time, t')
ylabel('sin(x), sin(x-1), sin(x-2)')
title('trei functii sinus')
```

### 10.2 Funcţii definite de utilizator

Spre deosebire de fişierele script prezentate anterior, funcţiile sunt fisiere *m* care acceptă parametri de intrare şi returnează unul sau mai multi parametri de ieşire.

Sintaxa unei funcţii este: (aceasta este şi prima linie din fişierul m):

function [parametri de iesire] = *nume_functie*(parametri de intrare)

Salvaţi fiecare funcţie în fişiere m separate. Numele fişierului trebuie să fie acelaşi cu numele functiei *nume_functie*.

O funcţie care calculează suma şi produsul a două numere se scrie astfel:

Listing 2: sum_product.m

```
function [sum, product] = sum_product(x,y)
%sum_product - returneaza suma si produsul a doua numere
% argumente de intrare:  x,y - valori scalare
% argumente de iesire:  sum  - suma lui x si y
%                       product: produsul lui x si y

sum = x+y;
product = x*y;

end
```

Funcţia poate fi apelată dintr-o altă funcţie, dintr-un fişier script sau din linia de comandă:

```
>> [s,p]=sum_product(2,3)
sau
>> [x,y]=sum_product(-4, 5)
```

## 11 Bucle şi instrucţiuni condiţionate

Limbajul Matlab are mai multe tipuri de bucle şi instrucţiuni condiţionate. Cele mai uzuale sunt:

- *for*:

  for index = valoare_initiala:increment:valoare_finala,
  instructiuni
  end

- *while*:

  while conditie,
  instructiuni
  end

- *if*:

  if conditie,
  instructiuni
  elseif conditie,
  instructiuni
  else
  instructiuni
  end

| Simbol | Descriere | Simbol | Descriere |
|---|---|---|---|
| < | Mai mic | & | SI logic |
| > | Mai mare | \| | SAU logic |
| == | Egal | ~ | NU logic |
| ~= | Diferit de | | |
| <= | Mai mic sau egal | | |
| >= | Mai mare sau egal | | |

Tabelul 7: Operatori relationali

In Tabelul 7 sunt ilustraţi cei mai utilizaţi operatori relaţionali. Pentru mai mulţi operatori, scrieti *help relop* în linia de comandă.

Fie polinomul:

$$p(x) = x^3 + 2x^2 + 3x + k$$

unde $k$ ia valori în $\{-2,\ -1,\ 0,\ 1,\ 2\}$. Următorul script calculează rădăcinile polinomului pentru fiecare valoare a lui $k$.

Listing 3: for_example.m

```
close all
clear all
clc
% radacinile unui polinom cu un coeficient variabil
for k = -2:2,               % k ia secvential valorile: -2, -1, 0, 1, 2
    polynomial = [1 2 3 k]; % se introduc coeficientii polinomului
    roots(polynomial)       % se calculeaza si se afiseaza radacinile in feresatra de comenzi
end;
```

Un exemplu de folosire a instrucţiunii *if* este cel din exemplul urmator. Se generează un număr $r$ aleator, cu valori cuprinse între 0 şi 1 şi se afişează un mesaj pentru cazul în care numărul este mai mic decât 0.5, între 0.5 şi 0.8, sau mai mare decat 0.8.

Listing 4: if_example.m

```
close all
clear all
clc
r = rand(1);    % se genereaza un numar aleator in intervalul (0,1)
disp('-------------------------------------------')
disp(r)         % se afiseaza numarul
if r < 0.5,
    disp('Numarul este mai mic decat 0.5');
elseif r <= 0.8
    disp('Numarul este in intervalul [0.5, 0.8]')
else
    disp('Numarul este mai mare decat 0.8')
end
disp('-------------------------------------------')
```

## 12 Exerciţii

**Exerciţiul 1:** Se consideră următoarele polinoame:

$$\begin{aligned} p_1(s) &= s^2(s+2) \\ p_2(s) &= (s+1)(s-7) \\ p_3(s) &= s^2 + 4 \\ p_4(s) &= s^2 + 3s + 2 \\ p_5(s) &= s^2 + 2s + 2 \\ p_6(s) &= s^4 - 16 \end{aligned}$$

i) Calculaţi rădăcinile polinoamelor (pe hârtie).

ii) Calculaţi rădăcinile polinoamelor în Matlab şi reprezentaţi-le grafic cu simbolul $*$, în figuri separate. Utilizati funcţiile *roots, plot, grid.*

**Exerciţiul 2:** Reprezentaţi grafic, în aceeaşi figură rădăcinile polinomului $p(x) = x^3 + 2x^2 + 3x + k$, pentru $k$: $0, 0.5, 1, 1.5, ..., 10$, folosind simbolul "*". Utilizaţi funcţiile: *for, roots, real, imag, plot, hold on, pause.*

**Exercitiul 3:** Calculaţi (pe hârtie şi utilizând Tabelul 8) transformarea Laplace inversă pentru:

$$\begin{aligned} F_1(s) &= \frac{1}{s+1} \\ F_2(s) &= \frac{1}{s^2+4} \\ F_3(s) &= \frac{1}{s-1} \\ F_4(s) &= \frac{1}{s^2-4} \\ F_5(s) &= \frac{1}{s^2+2s+5} \\ F_6(s) &= \frac{1}{s^2-4s+5} \end{aligned}$$

Reprezentaţi grafic funcţiile original $f_1(t), ..., f_6(t)$ pentru un interval de timp $t \in [0,\ 10]$, într-un script Matlab. Plasaţi graficele în acceaşi fereastră, în subfiguri separate aranjate într-o matrice cu 2 linii şi 3 coloane.

| $f(t)$ | $\mathcal{L}\{f(t)\}$ | $f(t)$ | $\mathcal{L}\{f(t)\}$ |
|---|---|---|---|
| $\delta(t)$ | $1$ | $\sin at$ | $\frac{a}{s^2+a^2}$ |
| $1$ | $\frac{1}{s}$ | $\cos at$ | $\frac{s}{s^2+a^2}$ |
| $t$ | $\frac{1}{s^2}$ | $e^{-at}\sin bt$ | $\frac{b}{(s+a)^2+b^2}$ |
| $e^{-at}$ | $\frac{1}{s+a}$ | $e^{-at}\cos bt$ | $\frac{s+a}{(s+a)^2+b^2}$ |

Tabelul 8: Tabel cu transformate Laplace

**Exerciţiul 4:** Scrieţi o funcţie *plotfreq(n)* care să afişeze, pe acelaşi grafic, următoarele funcţii, pentru intervalul de timp t = [0, 10].

$$f_1(t) = e^{-t}, \qquad f_2(t) = sin(2\pi nt), \qquad f_3(t) = f_1(t) \cdot f_2(t)$$

Parametrul de intrare este frecvenţa sinusului pentru funcţia ($f_2$). Apelaţi funcţia pentru mai multe valori ale lui $n \in [0.5,\ 5]$.

**Exerciţiul 5:** Se consideră polinoamele:

$$\begin{aligned} F_1(z) &= z^2 - 2az + 2a^2 \\ F_2(z) &= z^2 - az + a^2 \end{aligned}$$

unde $a$ este un parametru real.

i) Determinaţi intervalul de valori pentru $a$ astfel încât valoarea absolută a rădăcinilor să fie mai mică decât 1.

ii) Reprezentaţi grafic valoarea absolută a rădăcinilor în funcţie de intervalul rezultat pentru parametrul $a$ de la punctul i).

**Exerciţiul 6:** Scrieţi o funcţie care generează primele $n$ numere din şirul lui Fibonacci. Funcţia se va numi *fibonacci(n)* şi va returna vectorul: 1, 1, 2, 3, 5, ..., N.

$$F_1 = 1, \;\; F_2 = 1, \;\; ..., \;\; F_k = F_{k-1} + F_{k-2}$$

**Exerciţiul 7:** Desenaţi, în Matlab, un fractal *feriga*, [1]. Folosiţi următorul algoritm:

**Algorithm 1** Fractal ferigă

Definiti $numar\_de\_iteratii = 10000$
Definiti matricile:

$$A_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0.16 \end{bmatrix}, \; A_2 = \begin{bmatrix} 0.85 & 0.04 \\ -0.04 & 0.85 \end{bmatrix}, \; A_3 = \begin{bmatrix} 0.2 & -0.26 \\ 0.23 & 0.22 \end{bmatrix}, \; A_4 = \begin{bmatrix} -0.15 & 0.28 \\ 0.26 & 0.24 \end{bmatrix},$$

$$t_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \; t_2 = \begin{bmatrix} 0 \\ 1.6 \end{bmatrix}, \; t_3 = \begin{bmatrix} 0 \\ 1.6 \end{bmatrix}, \; t_4 = \begin{bmatrix} 0 \\ 0.44 \end{bmatrix}.$$

Setaţi punctul iniţial $(0, 0)$ ca vector coloană cu două elemente: $X = [0; 0]$. Acesta va fi prima coloană din matricea de coordonate $X$, unde fiecare coloană va conţine coordonatele unui punct care va fi reprezentat grafic.
Setaţi punctul curent egal cu punctul iniţial: $v = X$
**for** $i = 1$ to $numar\_de\_iteratii - 1$ **do**
  Generaţi un numar aleator $n$ intre 0 şi 1 (vezi funcţia Matlab *rand*)
  Calculaţi un punct nou $v$:
  **if** $n < 0.01$ **then**
    $v = A_1 \cdot v + t_1$
  **else**
    **if** $n < 0.8$ **then**
      $v = A_2 \cdot v + t_2$
    **else**
      **if** $n < 0.9$ **then**
        $v = A_3 \cdot v + t_3$
      **else**
        $v = A_4 \cdot v + t_4$
      **end if**
    **end if**
  **end if**
  Adăugaţi punctul curent $v$ la matricea de coordonate $X$ (în Matlab scrieţi: $X = [X \;\; v]$)
**end for**
Desenaţi elementele din a doua linie a matricii $X$ în funcţie de elementele de pe prima linie, cu simbolul verde "."

# Bibliografie


[1] Michael Barnsley. *Fractals Everywhere*. Academic Press, 1993.

[2] MathWorks. MATLAB, The Language of Technical Computing. `https://www.mathworks.com/help/matlab`, 2017.