INTRODUCERE ÎN MATLAB

OBIECTIVE

- Descrierea programului Matlab.
- Lucrul cu ferestre.
- Meniurile programului Matlab.
- Programarea în Matlab.
- Crearea fişierelor script şi function.
- Instrucțiuni şi funcții de control logic.
- Variabile speciale.

1.1. Introducere

Matlab este un pachet de programe de înaltă performanță, ce integrează calculul numeric şi reprezentările grafice în domeniile ştiinței şi ingineriei. Matlab oferă, pe baza unor lucrări matematice, o analiză matriceală, sinteza şi identificarea sistemelor şi programe grafice inginereşti atât 2D cât şi 3D. Matlab-ul integrează toate acestea într-un mediu uşor de învățat şi folosit, în care enunțurile problemelor şi rezolvările acestora sunt exprimate în modul cel mai natural posibil, aşa cum sunt scrise matematic, fără a fi necesară programarea tradițională.

O caracteristică importantă a acestui limbaj de programare este uşurința cu care acesta poate fi extins. Astfel, orice utilizator poate adăuga propriile programe scrise în Matlab la fişierele originale, dezvoltând aplicații specifice domeniului în care lucrează. Structural, Matlab-ul, este realizat sub forma unui nucleu de bază, cu interpretor propriu, în jurul căruia sunt construite toolbox-urile. Toolbox-urile sunt colecții extinse de funcții Matlab care dezvoltă mediul de programare de la o versiune la alta, pentru a rezolva probleme specifice anumitor domenii.

Firma The MathWorks Inc. (care a realizat acest limbaj) a pus în circulație o serie de toolbox-uri cum ar fi: Signal Processing, Image Processing, Symbolic Math, Neural Network, Control System Design, Robust Control, System Identification, Optimisation, Spline, Statistics, Wavelet, Simulink, Analysis and Synthesis, Finance, Fuzzy, etc.

1.2. Lansarea în execuție

Programul se lansează în execuție din mediul Windows, prin selecția pictogramei Matlab,


MATLAB 7.0.lnk

în momentul în care se lansează suprafața de lucru (desktop) a sistemului de operare (figura 1.2.1). Versiunea descrisă în această lucrare este Matlab 7 - Release 14.

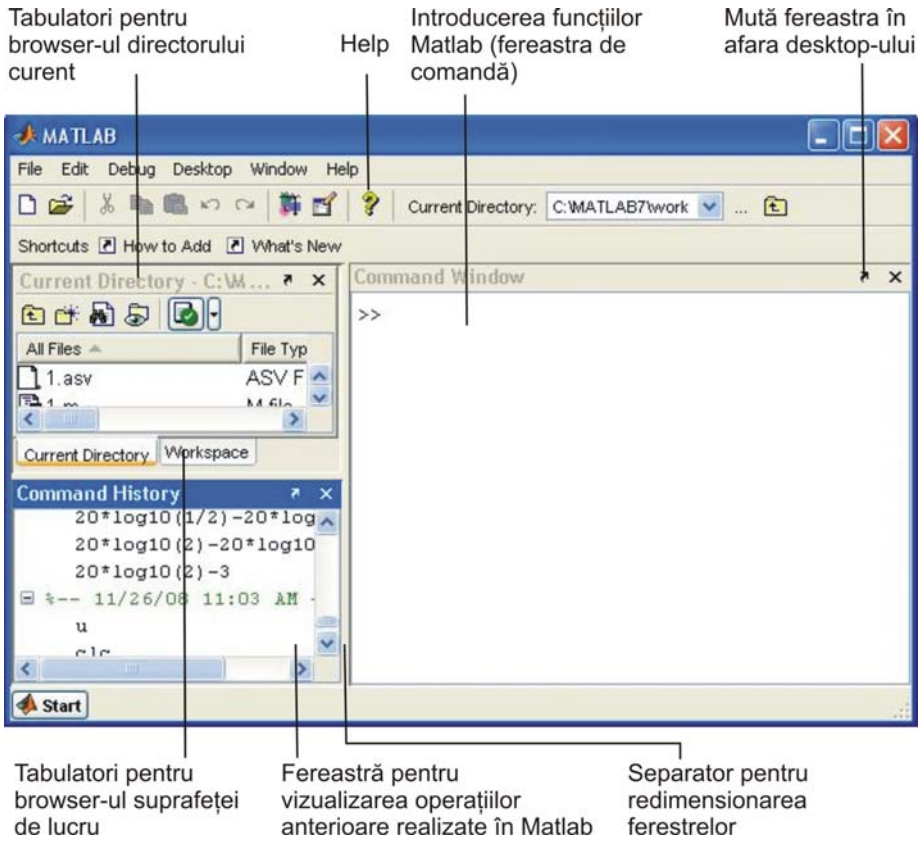

Fig. 1.2.1 Suprafața de lucru Matlab

Din afara mediului Windows (de exemplu DOS) programul poate fi lansat în execuție direct, cu o comandă de forma:

```
win c:\MATLAB\BIN\matlab
```

dacă fişierul „matlab.exe” se găseşte în calea „c:\MATLAB\BIN”.

1.3. Ferestrele de lucru

Programul Matlab lucrează cu două tipuri de ferestre: o fereastră de comenzi şi o fereastră pentru reprezentări grafice.

1.3.1. Fereastra de comenzi

Este utilizată pentru scrierea, modificarea şi depanarea funcțiilor programelor (fişierelor de tip M). Meniul din bara superioară este accesibil prin tastarea simultană a tastei [Alt] şi a literei subliniate a comenzii dorite sau prin selecția directă cu mouse-ul.

Fiecare comandă din meniul principal furnizează un meniu specific, selecția comenzii dorite făcându-se prin deplasarea zonei active cu ajutorul săgeților sau prin

selecția directă cu ajutorul mouse-lui.

Meniul **File** conține o serie de comenzi legate de fişiere:

- **New** – este folosit pentru deschiderea unui fişier nou de tip M (**M - file**), a unei noi ferestre grafice (**Figure**), a unui model Simulink nou (**Model**) sau a unei interfețe grafice de tip Graphical User Interface (**GUI**);
- **Open** – permite deschiderea unor fişiere deja existente pentru editare şi lansare;
- **Close Command Window** – închide fereastra de comandă Matlab;
- **Import Data** – opțiune destinată importului de date din fişiere;
- **Save Workspace As** – salvează toate variabilele din spațiul de lucru împreună cu valorile lor atribuite într-un fişier desemnat;
- **Set Path** – permite setarea căii fişierelor Matlab;
- **Preference** – accesează o casetă de dialog în care se pot seta o serie de caracteristici ale programului Matlab;
- **Print** şi **Print Selection** – tipăreşte conținutul ferestrei de comandă, respectiv a părții selectate;
- **Exit MATLAB** – opțiune folosită pentru părăsirea mediului Matlab.

Meniul **Edit** conține o serie de comenzi cunoscute din Windows destinate editării: **Undo**, **Redo**, **Cut**, **Copy**, **Paste**, **Paste Special**, **Select All**, **Delete**, precum şi trei comezi specifice pentru ştergerea conținutului ferestrei de comandă, ferestrei istoric şi suprafeței de lucru: **Clear Command Window**, **Clear Command History** şi **Clear Workspace**.

În meniul **View** se regăsesc comenzi destinate manipulării diferitelor ferestre aferente instrumentelor utilizate în Matlab. Comenzile din meniul **Web** pot accesa resursele din Internet ale companiei producătoare (pagina de web, suportul tehnic, etc.).

În meniul **Window** sunt afişate toate ferestrele deschise în timpul sesiunii de lucru curente, putându-se afişa fereastra activă. Ultimul meniu este **Help**, în care, prin intermediul comenzilor existente, se poate accesa documentația ataşată programului.

1.3.2. Fereastra de reprezentări grafice

Este o formă elevată de reprezentare a graficelor. Pot exista mai multe ferestre grafice deschise simultan, dar numai o singură fereastră de comenzi.

1.4. Programarea în Matlab

1.4.1. Matrice, vectori şi scalari

Matlab-ul lucrează numai cu un singur tip de obiecte, matrice numerice rectangulare, cu elemente reale sau complexe. În acest sens, scalarii sunt asimilați matricelor cu o linie şi o coloană (1x1), iar vectorii sunt asimilați matricelor cu o linie (1xN) sau cu o coloană (Nx1).

Elementele unei matrice pot fi identificate prin una dintre notațiile: A_{ij}, $A[i,j]$, $A(i,j)$, etc. şi semnifică elementul de la intersecția liniei i cu coloana j. Notația adoptată în Matlab este ultima dintre cele trei prezentate şi anume $A(i,j)$.

Dimensiunea unei matrice este precizată de o pereche de numere care arată numărul de linii şi coloane ale matricei respective.

Pentru a face referire la un element $A(i, j)$ al unei matrice A, sunt necesari doi indici, indicele de linie şi indicele de coloană, în această ordine. Referirea unui element al unui vector poate fi făcută numai cu un singur indice.

Definirea matricelor se poate face prin una din următoarele metode:

- introducerea explicită a listei de elemente;
- generarea prin instrucțiuni şi funcții;
- crearea de fişiere M[1];
- încărcarea din fişiere de date externe.

Matlab-ul nu conține instrucțiuni de dimensionare şi declarații de tip, iar memoria este alocată în mod automat până la valoarea maxim disponibilă. Cea mai simplă metodă de definire a matricelor mici constă în utilizarea unei liste explicite. La introducerea unei astfel de liste trebuie respectate următoarele reguli:

- elementele unei linii trebuie separate prin spațiu liber sau virgulă;
- liniile se separă prin semnul punct-virgulă "**;**";
- elementele unei matrici sunt cuprinse între paranteze drepte "**[ ]**".

1.4.2. Declarații şi variabile

Matlab-ul este un limbaj de expresii. Expresiile tipărite de utilizator sunt interpretate şi evaluate. Instrucțiunile Matlab sunt, de cele mai multe ori, de forma:

```
variabilă=expresie,
```

sau, mai simplu:

```
expresie.
```

Expresiile sunt compuse din operatori sau alte caractere speciale, din funcții şi nume de variabile. Evaluarea expresiei produce o matrice care este afişată pe ecran şi atribuită unei variabile. Dacă numele variabilei şi semnul egal („variabilă =") sunt omise, Matlab-ul crează automat o variabilă cu numele „**ans**".

Orice instrucțiune este în mod normal terminată cu „**Enter**". Dacă ultimul caracter al unei instrucțiuni este punct-virgulă „**;**", instrucțiunea este executată, dar tipărirea este suprimată. Utilizarea acestui caracter la sfârşitul unei instrucțiuni în fişiere M este necesară în situațiile în care nu se doreşte afişarea datelor intermediare. Numele de variabile şi de funcții au ca prim caracter o literă, urmată de litere, cifre sau caracterul special „liniuță de subliniere" (adică „**_**"). Matlab-ul face deosebirea între litere mari şi mici. Funcția "casesen" permite trecerea Matlab-ului în modul senzitiv/nesenzitiv, în vederea separării literelor mari de cele mici. La lansare, Matlab-ul este în modul senzitiv, iar cu comanda **casesen off** se trece în modul nesenzitiv. Revenirea se face cu comanda **casesen on**. Numele de funcții este obligatoriu să fie redate cu litere mici.

1.4.3. Structura programelor în Matlab

Matlab-ul lucrează fie în modul linie de comandă, caz în care fiecare linie este prelucrată imediat şi rezultatele sunt afişate, fie cu programe scrise în fişiere. Aceste

[1] Fişierele de tip M sunt fişiere ce conțin instrucțiuni Matlab şi se numesc aşa deoarece au extensia "m". Acestea sunt de fapt programe Matlab.

două moduri de lucru formează împreună un mediu de programare uşor de învăţat şi de utilizat. Fişierele ce conţin instrucţiuni Matlab se numesc fişiere M (deoarece au extensia “m”) şi sunt programe Matlab. Un fişier M constă dintr-o succesiune de instrucţiuni Matlab, cu posibilitatea apelării altor fişiere M precum şi a apelării recursive.

Un program Matlab poate fi scris sub forma unor fişiere „**script**” sau sub formă de fişiere „**function**”. Ambele tipuri de fişiere sunt scrise în format ASCII.

1.4.3.1. Fişierele script

Aceste fişiere sunt fişiere externe ce conţin secvenţe de instrucţiuni Matlab. Prin apelarea numelui fişierului, se execută secvenţa de instrucţiuni Matlab conţinută de acesta. După execuţia completă, variabilele cu care a operat fişierul rămân în zona de memorie a aplicaţiei. Fişierele script nu permit integrarea în programe mari realizate pe principiul modularizării, ele fiind folosite pentru rezolvarea unor probleme care cer comenzi succesive lungi, ce îngreunează lucrul în modul linie de comandă.

1.4.3.2. Fişierele funcţie (function)

Dacă prima linie a fişierului conţine cuvântul cheie „**function**”, fişierul respectiv este declarat ca fişier funcţie. Spre deosebire de un fişier script, un fişier funcţie poate lucra cu argumente. Variabilele definite şi utilizate în interiorul fişierului funcţie sunt localizate la nivelul acestuia. Deci, la terminarea execuţiei unei funcţii, în memorie rămân doar variabilele de ieşire ale funcţiei.

Forma generală a primei linii a unui fişier funcţie este următoarea:

```
function[param_ieşire]=nume_funcţie(param_intrare),
```

unde:

`function` - este cuvântul cheie care declară fişierul ca fişier funcţie; prezenţa acestui cuvânt cheie este obligatorie;
`nume_funcţie` - este numele funcţiei, adică numele sub care se salvează fişierul, fără extensie. Nu poate fi identic cu cel al unui fişier M preexistent.
`param_ieşire` - parametrii de ieşire trebuie separaţi cu virgulă şi cuprinşi între paranteze drepte (dacă funcţia nu are parametri de ieşire, parantezele drepte şi semnul egal nu mai au sens, pot să lipsească).
`param_intrare` - parametrii de intrare trebuie separaţi cu virgulă şi cuprinşi între paranteze rotunde (dacă funcţia nu are parametri de intrare, parantezele rotunde nu mai au sens, pot să lipsească).

1.4.4. Comentariile şi help-ul într-un program

Într-un program Matlab, un comentariu se introduce prin caracterul procent “**%**” plasat la începutul liniei. În cazul în care caracterul procent apare pe prima poziţie într-o linie, aceasta este omisă de compilator, iar dacă apare într-o linie de program, partea de linie care urmează după semnul procent va fi omisă.

Într-un program, un comentariu poate apărea în orice poziţie, dar este recomandată prezenţa unui comentariu după prima linie care declară o funcţie. În acest caz, comentariul, care apare imediat după prima linie de declarare a funcţiei, constituie help-ul fişierului respectiv. Dacă se apelează în fereastra de comandă:

```
help nume_funcţie,
```

se va afişa help-ul fişierului funcţiei respective.

1.5. Instrucțiuni şi funcții de control logic

Instrucțiunile de control logic în Matlab sunt: **if**, **else**, **elseif**, **for**, **while**, **break**, **return**, **end**, **error**.

1.5.1. Instrucțiunea condițională „if”

Această instrucțiune este folosită în cazul în care este necesară o selecție a grupului de instrucțiuni ce urmează a fi executat, selecție condiționată de valoarea de adevăr a unei expresii. Instrucțiunile condiționale utilizează operatorii relaționali şi operatorii logici.

Matlab-ul are şase operatori relaționali, care sunt utilizați pentru a compara două matrice de dimensiuni egale. Lista acestor operatori este prezentată în tabelul 1.5.1. Operatorii relaționali compară două matrice sau două expresii matriceale, element cu element. Rezultatul este o matrice de aceeaşi dimensiune cu a matricelor care se compară, cu elemente 1 când relația este ADEVĂRATĂ şi cu elemente 0 când relația este FALSĂ. Primii patru operatori compară numai partea reală a operatorilor, partea imaginară fiind ignorată, iar ultimii doi operatori tratează atât partea reală cât şi partea imaginară. Dacă unul dintre operanzi este un scalar şi celălalt este o matrice, scalarul se „extinde” până la dimensiunea matricei.

Forma generală de utilizare a acestor operatori este:

```
rezultat=expresie_1 operator_relațional expresie_2.
```

Tab. 1.5.1 Operatorii relaționali

Operatorii relaționali	Semnificația
<	mai mic
< =	mai mic sau egal
>	mai mare
> =	mai mare sau egal
= =	identic
~ =	diferit

Pentru combinarea a două sau mai multe expresii logice se utilizează operatorii logici prezentați în tabelul 1.5.2. Operatorii logici ŞI, SAU compară doi scalari sau două matrice de dimensiuni egale.

Tab. 1.5.2 Operatorii logici

Operatori logici	Simbol Matlab	Prioritatea
NU (not)	~	1
ŞI (and)	&	2
SAU (or)	|	3

Pentru cazul în care cei doi operanzi sunt matrice, se operează element cu element. Operatorul logic NU (sau complementul logic) este operator unar. Aceşti operatori au prioritate mai mică decât operatorii relaționali şi aritmetici.

Operatorii logici au prioritate mai mică decât operatorii relaționali sau cei aritmetici. În tabelul 1.5.3. este dată tabela de adevăr a operatorilor logici.

Tab. 1.5.3 Tabelul de adevăr al operatorilor logici

A	B	~A	A|B	A&B
FALS	FALS	ADEV.	FALS	FALS
FALS	ADEV.	ADEV.	ADEV.	FALS
ADEV.	FALS	FALS	ADEV.	FALS
ADEV.	ADEV.	FALS	ADEV.	ADEV.

Instrucțiunea if poate fi implementată ca instrucțiune **if** simplă, sau poate include clauzele **else** sau **elseif**. Forma generală a unei instrucțiuni **if** simplă este următoarea:

```
if  expresie logică
      grup de instrucţiuni
end
```

Dacă expresia logică este adevărată, se execută grupul de instrucțiuni cuprins între instrucțiunea **if** şi instrucțiunea **end**. În caz contrar, se trece la prima instrucțiune care urmează după instrucțiunea **end**.

Clauza **else** este folosită pentru a executa un set de instrucțiuni (grupul de instrucțiuni A) dacă expresia logică este adevărată şi un alt set de instrucțiuni (grupul de instrucțiuni B) dacă expresia logică este falsă.

Forma generală este:

```
if  expresie logică
    grup de instrucţiuni A
else
    grup de instrucţiuni B
end
```

Dacă funcția de calculat are mai multe nivele de instrucțiuni **if-else** este recomandată utilizarea clauzei **elseif**. Aceasta are următoarea formă generală:

```
if  expresie logică 1
    grup de instrucţiuni A
elseif  expresie logică 2
    grup de instrucţiuni B
elseif  expresie logică 3
    grup de instrucţiuni C
end
```

Această instrucțiune este evaluată în modul următor:

- dacă expresia logică 1 este adevărată, se execută numai grupul de instrucțiuni A;

- dacă expresia logică 1 este falsă şi expresia logică 2 este adevărată, se execută numai grupul de instrucțiuni B;
- dacă expresiile logice 1 şi 2 sunt false şi expresia logică 3 este adevărată, se execută numai grupul de instrucțiuni C;
- dacă mai multe expresii logice sunt adevărate, prima expresie logică adevărată determină care grup de instrucțiuni se execută prima dată;
- dacă nicio expresie logică nu este adevărată, nu se execută niciun grup de instrucțiuni din structura **if**.

Clauza **elseif** poate fi combinată cu clauza **else** într-o structură generală de forma:

```
if  expresie logică 1
    grup de instrucțiuni A
elseif  expresie logică 2
    grup de instrucțiuni B
elseif  expresie logică 3
    grup de instrucțiuni C
else
    grup de instrucțiuni D
end
```

În acest caz, dacă nicio expresie logică nu este adevărată, se execută grupul de instrucțiuni D.

1.5.2. Instrucțiunea repetitivă „for”

Instrucțiunea **for** permite repetarea unui grup de instrucțiuni din corpul buclei, de un anumit număr de ori. Structura generală a acestei instrucțiuni este următoarea:

```
for  index = expresie
    grup de instrucțiuni
end
```

unde:
`index` - este numele contorului;
`expresie` - este o matrice, un vector sau un scalar;
`grupul de instrucțiuni` - este orice expresie Matlab.

În aplicații, „**index = expresie**” are de cele mai multe ori următoarea formă:

```
k = inițial:pas:final
```

unde:
`inițial` - este prima valoare a variabilei k;
`pas` - este pasul, dacă acesta este omis, este considerat implicit 1;
`final` - este cea mai mare valoare pe care o poate lua variabila k.

La fiecare pas de calcul, „index” are valoarea unuia dintre elementele expresiei. Dacă expresia este o matrice, ciclarea se face pe coloane. Pentru un ciclu for cu pasul negativ sau neîntreg se generează mai întâi un vector cu pasul şi limitele dorite şi apoi se citesc valorile acestuia în cadrul buclei **for**. La folosirea buclei **for** trebuie respectate următoarele reguli:

- indexul buclei **for** trebuie să fie o variabilă;
- dacă expresia este o matrice goală, bucla nu se execută. Se va trece la următoarea instrucțiune după **end**;

- dacă expresia este un scalar, bucla se execută o singură dată cu indexul dat de valoarea scalarului;
- dacă expresia este un vector linie, bucla se execută de atâtea ori câte elemente are vectorul, de fiecare dată indexul având valoarea egală cu următorul element din vector;
- dacă expresia este o matrice, indexul va avea la fiecare iterație valorile conținute în următoarea coloană a matricei;
- la terminarea ciclului **for**, indexul are ultima valoare utilizată;
- dacă se utilizează operatorul două puncte „**:**” pentru a defini expresia, bucla se execută de $n = \left[\frac{final - initial}{pas} \right] + 1$ ori, dacă n este pozitiv, şi nu se execută dacă n este negativ. Prin **[ ]** s-a notat valoarea întreagă a numărului.

1.5.3. Instrucțiunea repetitivă „while”

Instrucțiunea repetitivă **while** este o structură care se utilizează pentru repetarea unui set de instrucțiuni, atât timp cât o condiție specificată este adevărată. Formatul general al acestei instrucțiuni este următorul:

```
while  expresie
    grup de instrucțiuni
end
```

Grupul de instrucțiuni este executat cât timp expresia este adevărată. Dacă expresia nu este adevărată, se trece la executarea primei instrucțiuni care urmează după **end**. Dacă expresia este întotdeauna adevărată logic, bucla devine infinită.

Notă. Dintr-o buclă infinită se iese forțat prin apăsarea concomitentă a tastelor [Ctrl]+[C] !

1.5.4. Instrucțiunea „break”

Instrucțiunea **break** se utilizează pentru a ieşi dintr-o buclă înainte ca aceasta să se fi terminat. Se recomandă utilizarea ei dacă o condiție de eroare este detectată în interiorul unei bucle. Această instrucțiune încetează execuția ciclurilor **for** şi **while**. În cazul unor cicluri imbricate, **break** comandă ieşirea din ciclul cel mai interior. Se apelează cu sintaxa: `break`.

1.5.5. Instrucțiunea „return”

Instrucțiunea **return** comandă o ieşire normală din fişierul M către funcția care l-a apelat sau către tastatură. Se apelează cu sintaxa: `return`.

1.5.6. Instrucțiunea „error”

Instrucțiunea **error** permite afişarea unor mesaje la întâlnirea unei erori. Se apelează cu sintaxa: `error('mesaj')`.

După afişarea textului „`mesaj`” controlul este redat tastaturii.

1.5.7. Funcții de control logic

În continuare, se vor prezenta pe scurt funcțiile de control logic. În Matlab există următoarele funcții de control logic:

exist - verifică dacă variabilele sau funcțiile argument sunt definite;
any - testează dacă cel puțin un element al unei matrice verifică o condiție logică dată (testează dacă un vector are cel puțin un element diferit de zero; returnează 1 dacă condiția este verificată şi 0 în caz contrar);
all - testează dacă toate elementele unei matrice verifică o condiție logică dată (testează dacă un vector are toate elementele diferite de zero; returnează 1 dacă condiția este adevarată şi 0 în caz contrar);
find - returnează indicii elementelor diferite de zero;
isnan - testează dacă elementele unei matrice sunt NaN[2];
isinf - testează dacă elementele unei matrice sunt infinite;
finite - testează dacă elementele unei matrice sunt finite.

Aceste funcții sunt des asociate cu instrucțiunea **if**.

1.6. Variabile speciale în Matlab

eps - variabilă în care este memorată variabila relativă, pentru calcule efectuate în virgulă mobilă, valoarea implicită fiind 2.2204e-016;
pi - variabilă permanentă ce are asociată valoarea 3.14159265358;
$i = j = \sqrt{-1}$ - variabilă folosită pentru reprezentarea unității imaginare;
inf - variabilă folosită pentru reprezentarea lui plus infinit în aritmetica IEEE;
nargin - variabilă permanentă folosită pentru testarea numărului argumentelor de intrare ce trebuie introduse pentru apelarea unei funcții;
nargout - variabilă permanentă folosită pentru testarea numărului argumentelor de ieşire ale unei funcții;
flops - returnează numărul de operații în virgulă mobilă efectuate de către calculator;
computer - variabilă folosită pentru obținerea informațiilor referitoare la tipul calculatorului şi numărul maxim de elemente pe care le poate gestiona versiunea respectivă de Matlab.
realmax - reprezintă cea mai mare valoare pozitivă în virgulă mobilă care poate fi folosită în calcule, respectiv 1.7977e+308;
realmin - reprezintă cea mai mică valoare pozitivă în virgulă mobilă care poate fi folosită în calcule, respectiv 2.2251e-308;
isieee - funcție ce returnează 1 dacă calculatorul este compatibil cu aritmetica IEEE şi respectiv 0 în caz contrar;
version, **ver** - funcții pentru determinarea versiunii Matlab şi a toolbox-urilor instalate pe calculator.

1.7. Exerciții propuse

Exercițiul 1.7.1

Să se introducă şi să se afişeze următoarele matrice:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}; \quad B = \begin{bmatrix} 1 & 3 & 5 & 7 \end{bmatrix}; \quad C = \begin{bmatrix} 1 \\ 4 \\ 9 \end{bmatrix};$$

[2] NaN – Not a Number, nu este un număr

Să se afişeze următoarele elemente:
- elementul de pe linia 2 şi coloana 1 din matrice;
- elementul al patrulea din vectorul linie;
- elementul al treilea din vectorul coloană.

Exercițiul 1.7.2

Să se scrie un fişier funcție, denumit „funcție1”, în care să se calculeze următoarea funcție:

$$f(x,y)=\begin{cases} x^3+y^3, & \textit{dacă} \quad 0 \le x-y \le 10 \\ x^2+y^2, & \textit{dacă} \quad x-y<0 \ \textit{şi} \ y \ge 0 \\ (x-y)^2, & \textit{în restul cazurilor} \end{cases}$$

Această funcție se va implementa în două moduri:

- Folosind instrucțiunea **if-else**.
- Folosind instrucțiunea **if-elseif**.

Să se calculeze următoarele valori: f(1,3), f(1,-3), f(-3,1), f(-4,-1).

Exercițiul 1.7.3

Fie $X=\begin{bmatrix} 1 & 0 & 2 \\ 0 & 0 & 1 \\ 0 & 0 & 4 \end{bmatrix}$

Să se determine:
- coloanele în care matricea are cel puțin un element diferit de zero;
- coloanele în care matricea are toate elementele diferite de zero;
- coloanele în care matricea are cel puțin un element mai mare ca –1.

Exercițiul 1.7.4

Să se calculeze suma primelor 100 de numere naturale. Se va calcula în două moduri: folosind instrucțiunea repetitivă **for** şi folosind instrucțiunea repetitivă **while**. Pentru rezolvare, se vor deschide două fişiere de tip M, în care se vor scrie cele două programe.

Exercițiul 1.7.5

Să se scrie un program care calculează suma elementelor vectorului $X=[5 \quad 2 \quad -9 \quad 10 \quad -1 \quad 9 \quad 1]$ până când întâlneşte un număr mai mare de 8.