2. ELEMENTE DE PROGRAMARE ÎN MATLAB

Obiectivele lucrării:

- familiarizarea cu tipurile de fişiere Matlab,
- recapitularea unor elemente de calcul vectorial,
- fixarea unor cunoştinţe privitoare la rezolvarea problemelor de calcul vectorial folosind mediul de programare Matlab,

prin studierea unor exemple şi prin rezolvarea unor probleme.

Se recomandă parcurgerea anexei M2 înaintea studierii paragrafelor 2.1 şi 2.2.

2.1. Elemente de programare în Matlab

Tipuri de fişiere Matlab

Fişierele Matlab (fişierele-M) sunt de două tipuri:

- *fişiere funcţie;*
- *fişiere script.*

Fişierele funcţie diferă de fişierele script prin faptul că primele pot lucra cu argumente (pot primi parametri şi returna valori), iar celelalte operează doar asupra variabilelor din *mediul de lucru* (*Workspace*). Fişierele script se utilizează de obicei pentru rezolvarea unor probleme care necesită executarea unui grup mare de comenzi, pentru care utilizarea modului linie de comandă ar fi greoaie.

Obligatoriu, prima linie a unui fişier funcţie este de forma:

```
function [parametri_ieşire]=nume_funcţie(parametri_intrare)
```

unde:

`function`	este cuvântul-cheie care declară fişierul ca fişier funcţie;
`nume_funcţie`	reprezintă numele funcţiei, este acelaşi cu numele sub care se salvează fişierul;
`parametri_ieşire`	reprezintă lista parametrilor de ieşire separaţi cu virgulă, cuprinsă între paranteze drepte; dacă funcţia nu are parametri de ieşire, atunci parantezele drepte şi semnul egal se omit;
`parametri_intrare`	reprezintă lista parametrilor de intrare separaţi cu virgulă, cuprinsă între paranteze rotunde; dacă funcţia nu are parametri de intrare, atunci parantezele rotunde se omit.

Rularea unui program de tip script (fişier script) se face fie tastând numele fişierului (fără extensie) în linia de comandă, fie apelând comanda *Run M-*

File a meniului *File*, fie apăsând tasta *F5* atunci când fişierul este deschis în editor şi editorul este fereastra activă.

Rularea unui program de tip funcţie (fişier funcţie) se face în linia de comandă, folosind sintaxa:

```
[variabile_valori_returnate]=nume_funcţie(valori_parametri_intrare)
```

unde:

`nume_funcţie`	numele funcţiei (fişierului în care este definită funcţia);
`variabile_valori_returnate`	lista variabilelor destinate memorării valorilor parametrilor de ieşire, separate cu virgulă, cuprinsă între paranteze drepte; dacă funcţia nu are parametri de ieşire, atunci parantezele drepte şi semnul egal se omit; dacă funcţia are un singur parametru de ieşire, parantezele drepte se omit;
`valori_parametri_intrare`	lista valorilor parametrilor de intrare separate cu virgulă, cuprinsă între paranteze rotunde; dacă funcţia nu are parametri de intrare, atunci parantezele rotunde se omit.

Comentarii în Matlab

Un *comentariu* într-un program Matlab are forma:

```
%comentariu
```

Dacă pe o linie de program apare caracterul %, atunci partea de linie care urmează după acest caracter va fi omisă de către interpretor. Dacă un comentariu se întinde pe mai multe linii, fiecare dintre aceste linii trebuie să aibă la început caracterul %.

Dacă un comentariu apare imediat sub linia de declarare a unui fişier funcţie, el va constitui *help*-ul fişierului funcţie respectiv, adică textul care va fi afişat în linia de comandă în urma apelului:

```
>> help nume_functie
```

unde `nume_functie` este numele (fără extensie) a fişierului funcţie.

Există şi o formă specială a unui *comentariu pe mai multe linii*, numită *block comment*, comentariul fiind delimitat de două linii speciale. Comentariul propriu-zis se scrie pe linii cuprinse între cele două linii delimitatoare:

```
%{
comentariu
%}
```

Instrucţiuni de control

Tabelele 2.1 şi 2.2 conţin ***operatorii relaţionali*** şi, respectiv, ***operatorii logici*** utilizaţi în Matlab:

Tabelul 2.1. *Operatori relaționali utilizați în Matlab.*

Operatori relaţionali	Semnifi-caţie
<	strict mai mic
>	strict mai mare
<=	mai mic sau egal
>=	mai mare sau egal
==	identic
~=	diferit

Tabelul 2.2. *Operatori logici utilizați în Matlab.*

Operatori logici	Semnifi-caţie	Prioritate
~	NU	1
&	ŞI	2
|	SAU	3

Comentarii: 1. În cazul folosirii operatorilor relaționali `<`, `>`, `<=`, `>=` pe mulțimea matricelor de numere complexe, se constată că este omisă partea imaginară. Astfel, `0>=i` va da ca rezultat răspunsul 1 (adică TRUE) (`0>=0`, unde al doilea `0` este partea reală a lui `i`).

2. Operatorii logici se utilizează întotdeauna pentru compararea matricelor cu elemente 0 şi 1 (având semnificațiile fals, respectiv, adevărat), calculate cu operatori relaționali.

Instrucțiunea "if" poate fi folosită în una din următoarele trei forme, având sintaxele:

- instrucțiunea *if* simplă:

```
if expresie_logică
     grup_de_instrucțiuni
end
```

- clauza *else*:

```
if expresie_logică
     grupul_de_instrucțiuni_1
else
     grupul_de_instrucțiuni_2
end
```

- clauza *elseif* :

```
if expresia_logică_1
     grupul_de_instrucțiuni_1
elseif  expresia_logică_2
     grupul_de_instrucțiuni_2
...
elseif  expresia_logică_n-1
     grupul_de_instrucțiuni_n-1
else
     grupul_de_instrucțiuni_n
end
```

Sintaxa instrucțiunii repetitive "for" din Matlab este de forma:

```
for index=expresie
     grup_de_instrucțiuni
end
```

în care:

- *expresie* este o matrice, un vector sau un scalar; de cele mai multe ori este de forma:

```
k=inițial:pas:final
```

- dacă *expresie* este o matrice, atunci ciclarea se face pe coloane.

Instrucțiunea repetitivă "while" este utilizată pentru repetarea unui set de instrucțiuni, atâta timp cât o condiție specificată este adevărată. Sintaxa acestei instrucțiuni este de forma:

```
while expresie
       grup_de_instrucțiuni
end
```

cu următoarele comentarii:

- *grupul_de_instrucțiuni* se execută cât timp toate elementele din *expresie* sunt nenule;
- *expresie* este de obicei sub forma:

```
expresie_1 operator_relațional expresie_2
```

- ieşirea forțată dintr-o buclă infinită se face prin apăsarea concomitentă a tastelor [Ctrl] şi [C].

Funții de testare

În tabelul 2.3 sunt redate pe scurt o parte din funțiile folosite pentru testări de valori ale variabilelor, a tipurilor variabilelor, obiectelor etc:

Tabelul 2.3. *Funcții Matlab de testare.*

Funcţia	**Efectul**
ischar	testează dacă argumentul dat este un şir de caractere
isempty	testează dacă argumentul dat este matricea vidă (matricea fără niciun element)
isequal	testează dacă două tablouri sunt identice
isfinite	găseşte elementele finite ale unui tablou
isfloat	testează dacă argumentul dat este un tablou de valori în virgulă mobilă
isinf	găseşte elementele de valoare infinită ale unui tablou (Inf)
isinteger	testează dacă argumentul dat este un tablou de întregi
iskeyword	testează dacă argumentul dat este un cuvânt-cheie Matlab

Tabelul 2.3. *Funcții Matlab de testare - continuare*

Funcţia	**Efectul**
isnan	găseşte elementele unui tablou care nu sunt numere (NaN)
isnumeric	testează dacă argumentul dat este un tablou de numere
isprime	găseşte elementele unui tablou care sunt numere prime
isreal	testează dacă toate elementele unui tablou sunt numere reale (coeficientul părții imaginare este zero)
isscalar	testează dacă argumentul dat este un scalar
issorted	testează dată un set de elemente este ordonat crescător
isvector	testează dacă argumentul dat este un vector

Calcul vectorial

În tabelul 2.4 sunt redate principalele funcții utilizate în ***calculul vectorial***.

Tabelul 2.4. *Funcții Matlab pentru calcul vectorial.*

Funcţia	**Sintaxa**	**Efectul**
norm	`norm(v,p)`	returnează norma `p` a vectorului `v`
	`norm(v)`	returnează norma euclidiană a vectorului `v` (identic cu cazul `p=2`)
dot	`dot(v,w)`	returnează produsul scalar al vectorilor de aceeaşi lungime, `v` şi `w`
cross	`cross(v,w)`	returnează produsul vectorial al vectorilor cu câte 3 elemente, `v` şi `w`
sum	`sum(v)`	returnează suma elementelor vectorului `v`
length	`length(v)`	returnează lungimea vectorului `v` (adică numărul de elemente)

2.2. Exemple de studiat

Exemplul 2.1: Să se genereze matricele pătratice A şi B, de ordinul 4, definite prin relațiile de mai jos. Apoi să se afişeze suma lor, produsul lor, cubul matricei A, rezultatul împărțirii la stânga a matricei A prin B şi rangul matricei B. Rezolvarea problemei să se facă prin utilizarea unui fişier script.

$$A_{i,j} = \frac{1}{i+j}, \quad i,j = \overline{1,4}$$

$$B_{i,j}=\begin{cases}1, & i=j\\ i+j, & i>j\\ i-j, & i<j\end{cases},\quad i,j=\overline{1,4}$$

Soluție: Etapele rezolvării problemei sunt următoarele:

i) Crearea unui fişier-M, numit, de exemplu, `op_matr.m`:

```
>> edit op_matr
```

ii) Rezolvarea cerințelor problemei prin scrierea comenzilor în fişierul script:

```
% generarea matricei A
for i=1:4
    for j=1:4
        A(i,j)=1/(i+j);
    end
end
% generarea matricei B
for i=1:4
    for j=1:4
        if i==j B(i,j)=1;
        elseif i>j B(i,j)=i+j;
        else B(i,j)=i-j;
        end
    end
end
% afisarea matricei A
A
% afisarea matricei B
disp('matricea B')
disp(B)
% calculul si afisarea sumei
Suma=A+B
% calculul si afisarea produsului
Produs=A*B
% calculul si afisarea cubului matricei A
cub_A=A^3
% calculul si afisarea rezultatului impartirii la stanga
disp('Rezultatul impartirii A\B este')
Rez=A\B
% rangul matricei B
rang_B=rank(B)
```

Se recomandă salvarea constantă a fişierului.

iii) Rularea fişierului script se face din linia de comandă:

```
>> op_matr
```

Rularea unui fişier-M din linia de comandă are ca efect rularea ultimelei versiuni salvate a fişierului.

iv) Vizualizarea rezultatelor afişate în fereastra de comenzi:

```
A =
    0.5000    0.3333    0.2500    0.2000
    0.3333    0.2500    0.2000    0.1667
    0.2500    0.2000    0.1667    0.1429
    0.2000    0.1667    0.1429    0.1250
matricea B
```

```
     1    -1    -2    -3
     3     1    -1    -2
     4     5     1    -1
     5     6     7     1
Suma =
    1.5000   -0.6667   -1.7500   -2.8000
    3.3333    1.2500   -0.8000   -1.8333
    4.2500    5.2000    1.1667   -0.8571
    5.2000    6.1667    7.1429    1.1250
Produs =
    3.5000    2.2833    0.3167   -2.2167
    2.7167    1.9167    0.4500   -1.5333
    2.2310    1.6405    0.4667   -1.1738
    1.8964    1.4310    0.4512   -0.9512
cub_A =
    0.4516    0.3263    0.2571    0.2127
    0.3263    0.2358    0.1859    0.1538
    0.2571    0.1859    0.1465    0.1213
    0.2127    0.1538    0.1213    0.1004
Rezultatul impartirii A\B este
Rez =
  1.0e+004 *
   -0.0600    0.2380   -0.4940   -0.1420
    0.4620   -1.5900    3.7980    1.0920
   -0.9660    2.8980   -7.7280   -2.2260
    0.5880   -1.5680    4.5640    1.3160
rang_B =     4
```

Exemplul 2.2: Să se scrie o funcție Matlab care primeşte ca argumente trei valori a,b,p şi care generează vectorul cu pas liniar v=a:p:b. Funcția returnează vectorul generat, lungimea sa, precum şi suma elementelor sale. Să se testeze funcția creată pentru următoarele triplete de valori: (0,25,5), (2,19,3), (5,-3,-2), (5,-2,0), (2,19,-1).

Soluție: Etapele rezolvării problemei sunt următoarele:

i) Crearea unui fişier-M, numit, de exemplu, `vect_lin.m`:

```
>> edit vect_lin
```

ii) Rezolvarea cerințelor problemei prin scrierea comenzilor în fişierul funcție:

```
function [v,lung,suma]=vect_lin(a,b,p)
v=a:p:b; lung=length(v);
if lung>0
    suma=sum(v);
else
    suma=[];
end
```

iii) Rularea fişierului funcție pentru tripletele de valori cerute şi vizualizarea rezultatelor:

```
>> [v,lung,suma]=vect_lin(0,25,5)

v =
     0     5    10    15    20    25
```

```
lung =     6
suma =    75
>> amin=2; amax=19; pas=3;
>> [w,l,s]=vect_lin(amin,amax,pas)
w =
     2     5     8    11    14    17
l =     6
s =    57
>> a=5; b=-3; p=-2;
>> [v,lung,suma]=vect_lin(a,b,p)
v =
     5     3     1    -1    -3
lung =     5
suma =     5
>> [v1,l1,s1]=vect_lin(5,-2,0)
v1 =
   Empty matrix: 1-by-0
l1 =     0
s1 =     []
>> [v2,l2,s2]=vect_lin(2,19,-1)
v2 =
   Empty matrix: 1-by-0
l2 =     0
s2 =     []
```

Exemplul 2.3: Să se scrie o funcție Matlab care primeşte ca argumente doi vectori, v şi w, de lungimi egale cu 3, şi returnează normele euclidiene ale vectorilor, produsul lor scalar, produsul lor vectorial şi unghiul dintre cei doi vectori exprimat în radiani.

Soluție: Etapele rezolvării problemei sunt următoarele:

i) Crearea fişierului-M:

```
>> edit vectori
```

ii) Scrierea comenzilor în fişierul-M creat:

```
function [n_v,n_w,ps,pv,unghi]=vectori(v,w)
if length(v)~=3 | length(w)~=3
    disp('Vectorii nu satisfac conditia de lungime 3!')
    n_v=[]; n_w=[]; ps=[]; pv=[]; unghi=[];
    return;
end
n_v=norm(v); n_w=norm(w);
ps=dot(v,w); pv=cross(v,w);
if n_v==0 | n_w==0
    disp('Unghiul nu poate fi calculat, unul din vectori fiind
zero.')
    unghi=[];
else
    unghi=acos(ps/(n_v*n_w));
end
```

iii) Testarea programului pentru diferite perechi de vectori:

```
>> v=[1 -1 3]; w=[0 3 -2];
>> [n_v,n_w,ps,pv,unghi]=vectori(v,w)
```

```
n_v =    3.3166
n_w =    3.6056
ps =    -9
pv =    -7     2     3
unghi =   2.4228
>> a=[1 2 3]; b=[0 0 0]; c=[-1 -2];
>> [n_v,n_w,ps,pv,unghi]=vectori(a,b)
Unghiul nu poate fi calculat, unul din vectori fiind zero.
n_v =    3.7417
n_w =     0
ps =     0
pv =     0     0     0
unghi =     []
>> [n_v,n_w,ps,pv,unghi]=vectori(a,c)
Vectorii nu satisfac conditia de lungime 3!
n_v =     []
n_w =     []
ps =     []
pv =     []
unghi =     []
```

Comentarii: 1. Este necesară verificarea respectării condițiilor impuse argumentelor transmise funcției (în acest caz, faptul ca cei doi vectori transmişi ca parametri să aibă fiecare lungimea egală cu 3).

2. La testare trebuie considerate atât cazul general, cât şi cazurile particulare care generează erori.

3. Comanda `return` determină oprirea execuției programului şi returnarea controlului către funcția apelantă sau către tastatură.

Exemplul 2.4: Pentru o matrice pătratică transmisă ca parametru unei funcții Matlab se cere să se scrie o secvență de instrucțiuni prin intermediul căreia: să se specifice dacă matricea este inversabilă; în cazul în care matricea este inversabilă, să se afişeze inversa ei, iar în cazul în care matricea nu este inversabilă, să se afişeze rangul ei.

Soluție: Rezolvarea constă din următoarele etape:

i) Crearea fişierului-M:

```
>> edit calcul_matr
```

ii) Scrierea codului sursă în fişier:

```
function calcul_matr(M)
[lin,col]=size(M);
if lin~=col
    disp('Matricea nu este patratica!')
    return
end
if det(M)~=0
    disp('Matricea este inversabila.')
```

```
    inversa=inv(M)
else
    disp('Matricea nu este inversabila.')
    rang=rank(M)
end
```

iii) Testarea programului:

```
>> A=[1 0; -1 1]; B=[1 1; 2 2]; C=[1 1 2; 2 2 3];
>> calcul_matr(A)
Matricea este inversabila.
inversa =
     1     0
     1     1
>> calcul_matr(B)
Matricea nu este inversabila.
rang =     1
>> calcul_matr(C)
Matricea nu este patratica!
```

Exemplul 2.5: Să se definească în Matlab funcția f:$\boldsymbol{R}\rightarrow\boldsymbol{R}$, $f(x)=\frac{1}{1+e^{-3\cdot x}}$.

Soluție: Rezolvarea constă din următoarele etape:

i) Crearea fişierului-M:

```
>> edit f
```

ii) Scrierea codului sursă în fişier:

```
function y=f(x)
if imag(x)==0
    y=1./(1+exp(-3*x));
else
    y='x trebuie sa fie numar real sau vector de numere reale';
end
```

iii) Testarea programului:

```
>> f(0.5)
ans =    0.8176
>> y=f(-2)
y =    0.0025
>> f(1+i)
x trebuie sa fie numar real sau vector de numere reale
```

Comentarii: 1. Funcția Matlab `imag` returnează coeficientul părții imaginare al unui număr complex.

2. Matlab lucrează implicit cu numere complexe. Funcția matematică fiind definită pe un domeniu de numere reale, este necesar să se testeze dacă argumentul primit de funcția Matlab corespunzătoare este un număr real sau un vector de numere reale.

3. Comparația `imag(x)==0` poate fi înlocuită cu `isreal(x)==1`.

4. Este recomandabil ca în cazul funcțiilor să se folosească operatori pentru tablouri, pentru a putea calcula simultan valorile funcției în mai multe puncte ale domeniului de definiție.

5. Dacă se doreşte calculul simultan al valorilor funcției f în mai multe puncte ale domeniului de definiție, atunci se procedează în felul următor:

- se creează un vector cu punctele din domeniul de definiție în care se doreşte aflarea valorilor funcției f, de exemplu:

```
>> x=[-5.3 -2 -1.47 0 0.25 1];
```

- se apelează fişierul funcție care conține definiția funcției f:

```
>> y=f(x)
```

- ca rezultat al acestui apel este afişarea valorilor funcției f în punctele dorite:

```
y =    0.0000    0.0025    0.0120    0.5000    0.6792    0.9526
```

2.3. Probleme de rezolvat

P2.1. Să se genereze şi să se afişeze:

- matricea pătratică de ordinul n=4, ale cărei elemente sunt date de expresia:

$$M_{i,j} = \frac{i \cdot j}{i + j - 1}, \quad i, j = \overline{1,n};$$

- o matrice A cu 4 linii şi 5 coloane, ale cărei elemente sunt:

$$A_{i,j} = \begin{cases} 3, & dacă\ i = j \\ -3, & dacă\ |i - j| = 2 \\ 1, & dacă\ i + j = 3 \\ 0, & în\ rest \end{cases}, \quad i = \overline{1,4},\ j = \overline{1,5}.$$

P2.2. Să se scrie un program, care, utilizând o instrucțiune *while*, calculează produsele parțiale ale elementelor vectorului $v1$=[2 3 1 9 2 -1 -3 5] până când întâlneşte un număr strict negativ şi afişează ultimul produs calculat. Care este rezultatul afişat, dacă se înlocuieşte vectorul $v1$ cu vectorul $v2$=[2 3 1 9 2 1 3 5]?

P2.3. Să se scrie o funcție Matlab care primeşte ca argument o matrice cu cel puțin 4 linii şi 4 coloane şi care afişează rezultatele următoarelor operații de extragere de elemente ale matricei:

- linia a 3-a;
- ultima coloană;
- ultima linie;

- submatricea determinată de liniile 2-4 şi coloanele 1-3.

P2.4. Fie vectorii:

$$\vec{v} = 2\vec{i} - \vec{j} + 3\vec{k}, \qquad \vec{w} = 3\vec{j} - 2\vec{k}$$

Să se determine, scriind un program Matlab, normele euclidiene, produsul scalar, cosinusul unghiului, unghiul exprimat în grade şi produsul vectorial pentru cei doi vectori.

P2.5. Să se scrie o funcție Matlab care primeşte ca argument o matrice pătratică şi afişează transpusa, rangul şi determinantul matricei.

P2.6. Să se scrie o funcție Matlab care primeşte ca argument o matrice pătratică nesingulară şi care returnează determinantul şi inversa matricei.

P2.7. Să se definească în Matlab funcția f:$\boldsymbol{R}\rightarrow\boldsymbol{R}$, $f(x)=\begin{cases}\dfrac{\sin(3\cdot x)}{2\cdot x}, & x<0\\ \cos(3\cdot x), & x\geq 0\end{cases}$. Să se calculeze şi să se afişeze valorile funcției în punctele -3, $-\dfrac{\pi}{2}$, 0, 1.25, $\dfrac{7\cdot\pi}{2}$.

2.4. Întrebări recapitulative

Î2.1. Precizați cum se poate ieşi forțat dintr-o buclă infinită.

Î2.2. Ce se înțelege prin „norma euclidiană” a unui vector?

Î2.3. Precizați la modul general cum se calculează produsul scalar a doi vectori cunoscuți prin proiecțiile lor pe axele sistemului de coordonate.

Î2.4. Precizați la modul general cum se calculează produsul vectorial a doi vectori cunoscuți prin proiecțiile lor pe axele sistemului de coordonate.

Î2.5. Definiți noțiunea de „produs vectorial a doi vectori”.

Î2.6. Definiți noțiunea de „unghi dintre doi vectori”.

Î2.7. Precizați care este funcția Matlab pentru determinarea lungimii unui vector, precum şi modul ei de apelare.

ANEXA M2. ELEMENTE DE ALGEBRĂ VECTORIALĂ

M2.1. Elemente de algebră vectorială

Se consideră spațiul real *n*-dimensional, raportat la un sistem ortogonal de coordonate având versorii $\vec{i}_1, \vec{i}_2, \ldots, \vec{i}_n$.

Fie $\vec{v}$ şi $\vec{w}$ doi vectori cunoscuți prin proiecțiile lor pe axele sistemului de coordonate:

$$\vec{v} = v_1 \cdot \vec{i}_1 + v_2 \cdot \vec{i}_2 + \cdots + v_n \cdot \vec{i}_n$$

$$\vec{w} = w_1 \cdot \vec{i}_1 + w_2 \cdot \vec{i}_2 + \cdots + w_n \cdot \vec{i}_n$$

Norma euclidiană a (***modulul, mărimea***) ***vectorului*** $\vec{v}$ este numărul real pozitiv determinat prin relația:

$$v = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}$$

Un vector al cărui modul este egal cu 1 se numeşte ***vector unitar*** sau ***versor***.

Norma p a vectorului $\vec{v}$ este numărul real pozitiv determinat prin relația:

$$v_p = \sqrt[p]{|v_1|^p + |v_2|^p + \ldots + |v_n|^p}, \quad p \in [1, \infty)$$

Produsul scalar al vectorilor $\vec{v}$ ***şi*** $\vec{w}$ este un număr (scalar) real determinat prin relația:

$$\vec{v} \cdot \vec{w} = v_1 \cdot w_1 + v_2 \cdot w_2 + \cdots v_n \cdot w_n$$

Vectorii $\vec{v}$ şi $\vec{w}$ sunt ***ortogonali*** dacă şi numai dacă $\vec{v} \cdot \vec{w}$=0.

Prin ***unghiul dintre vectorii*** $\vec{v}$ ***şi*** $\vec{w}$ se înțelege unghiul mic determinat de sensurile pozitive ale celor doi vectori. Unghiul dintre vectorii $\vec{v}$ şi $\vec{w}$ se notează cu $\angle(\vec{v}, \vec{w})$. Cosinusul acestui unghi se calculează cu formula:

$$cos(\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{v \cdot w}$$

În cazul spațiului tridimensional (*n*=3), se defineşte ***produsul vectorial al vectorilor*** $\vec{v}$ ***şi*** $\vec{w}$, în cazul în care aceştia sunt necoliniari şi nenuli, ca fiind vectorul notat $\vec{v} \times \vec{w}$ care are direcția perpendiculară pe direcțiile celor doi vectori, sensul dat de aplicarea regulii burghiului drept când se roteşte primul vetor ($\vec{v}$) peste cel de-al doilea vector ($\vec{w}$), şi modulul egal cu produsul $v \cdot w \cdot \sin(\vec{v}, \vec{w})$, respectiv vectorul zero,

în cazul în care cei doi vectori sunt coliniari sau cel puțin unul dintre ei este vectorul nul.

Produsul vectorial al vectorilor $\vec{v}$ şi $\vec{w}$ se poate calcula cu formula:

$$\vec{v}\times\vec{w}=\begin{vmatrix}\vec{i}_1 & \vec{i}_2 & \vec{i}_3\\ v_1 & v_2 & v_3\\ w_1 & w_2 & w_3\end{vmatrix}=(v_2w_3-v_3w_2)\vec{i}_1+(v_3w_1-v_1w_3)\vec{i}_2+(v_1w_2-v_2w_1)\vec{i}_3$$

Doi vectori nenuli şi necoliniari $\vec{v}$ şi $\vec{w}$ sunt ***paraleli*** dacă şi numai dacă $\vec{v}\times\vec{w}=0$.