

Cuprins

Matrice în MATLAB.....	1
Generarea matricelor.....	1
Indexarea și notația ":"	11
Operații în sens matricial și în sens tablou	13
Analiza datelor	19
Operatori relaționali și logici	21

Matrice în MATLAB

Matricele sunt tipuri de date fundamentale în MATLAB. Ele sunt de fapt tablouri multidimensionale în dublă precizie. Cele mai folosite sunt matricele bidimensionale, care sunt tablouri bidimensionale cu m linii și n coloane. Vectorii linie ($m = 1$) și coloană ($n = 1$) sunt cazuri particulare de matrice bidimensionale.

Generarea matricelor

Matricele pot fi generate **explicit**. Delimitatorii: [], blanc sau virgula în interiorul liniei, ; sau CR între linii.

```
A = [5 7 9  
1 -3 -7]
```

```
A =  
      5      7      9  
      1     -3     -7
```

Echivalent

```
A = [5, 7, 9; 1, -3, -7]
```

```
A =  
      5      7      9  
      1     -3     -7
```

Dimensiunea unei matrice se obține cu comanda `size`

```
v = size(A)
```

```
v =  
      2      3  
[r,c] = size(A)
```

```
r =
```

```
2
```

```
c =
```

```
3
```

Generare de matrice speciale

<code>zeros</code>	Matricea nulă
<code>ones</code>	Matrice formata numai din elemente 1
<code>eye</code>	Matricea unitate
<code>repmat</code>	Replicarea matricelor
<code>rand</code>	Matrice aleatoare cu elemente uniforme in $[0,1]$
<code>randn</code>	Matrice aleatoare cu elemente distribuite normal
<code>linspace</code>	Vector de elemente echidistante
<code>logspace</code>	Vector de elemente spațiate logaritmice

Primele șase au aceeași sintaxă, de exemplu `zeros(m,n)` sau `zeros([m,n])` generează matricea nulă de tip $m \times n$. `eye(size(A))` generează o matrice unitate de aceeași dimensiune ca A. `ones(n)` este echivalentă cu `ones(n,n)`.

rand

```
ans =  
    0.1576
```

rand(3)

```
ans =
```

```

0.9706    0.8003    0.9157
0.9572    0.1419    0.7922
0.4854    0.4218    0.9595

```

Reproductibilitatea experimentelor – inițializarea generatorului `rand('state', j)`. Inițializează generatorul la starea `j`. `j=0` este starea implicită de la lansarea MATLAB.

Generare pe blocuri

Fie matricea B

```
B = [1, 2; 3, 4]
```

```
B =
```

```

1    2
3    4

```

Din ea generăm:

```
C=[B, zeros(2); ones(2), eye(2)]
```

```
C =
```

```

1    2    0    0
3    4    0    0
1    1    1    0
1    1    0    1

```

`blkdiag` permite generarea matricelor diagonale pe blocuri

```
A=blkdiag(2*eye(2),ones(2))
```

```
A =
```

```

     2     0     0     0
     0     2     0     0
     0     0     1     1
     0     0     1     1

```

Funcția `repmat` permite construirea de matrice prin repetarea de subblocuri: `repmat(A,m,n)` crează o matrice de m pe n blocuri în care fiecare bloc este o copie a lui A . Dacă n lipsește, valoarea sa implicită este m . Exemplu:

```
A=repmat(eye(2),2)
```

```
A =
```

```

     1     0     1     0
     0     1     0     1
     1     0     1     0
     0     1     0     1

```

Tabela următoare dă câteva funcții pentru manipularea matricelor.

<code>reshape</code>	Schimbarea dimensiunii
<code>diag</code>	Matrice diagonale și diagonale ale matricelor
<code>blkdiag</code>	Matrice diagonală pe blocuri

<code>tril</code>	Extragerea părții triunghiulare inferior
<code>triu</code>	Extragerea părții triunghiulare superior
<code>fliplr</code>	Rotire matrice în jurul axei de simetrie verticale
<code>flipud</code>	Rotire matrice în jurul axei de simetrie orizontale
<code>rot90</code>	Rotația unei matrice cu 90 de grade

Funcția `reshape` schimbă dimensiunile unei matrice: `reshape(A, m, n)` produce o matrice `m` pe `n` ale cărei elemente sunt luate coloană cu coloană din `A`. De exemplu:

`A=[1 4 9; 16 25 36], B=reshape(A,3,2)`

`A =`

```

     1     4     9
    16    25    36

```

`B =`

```

     1    25
    16     9
     4    36

```

Funcția `diag` lucrează cu diagonalele unei matrice și poate avea ca argument o matrice sau un vector. Pentru un vector `x`, `diag(x)` este matricea cu diagonala principală `x`:

`diag([1,2,3])`

`ans =`

```

     1     0     0

```

```

0      2      0
0      0      3

```

Mai general, `diag(x, k)` pune `x` pe diagonală cu numărul `k`, unde `k = 0` înseamnă diagonală principală, `k > 0` specifică diagonale situate deasupra diagonalei principale, iar `k < 0` diagonale dedesubtul diagonalei principale:

```
diag([1,2],1)
```

```
ans =
```

```

0      1      0
0      0      2
0      0      0

```

```
diag([3 4],-2)
```

```
ans =
```

```

0      0      0      0
0      0      0      0
3      0      0      0
0      4      0      0

```

Pentru o matrice `A`, `diag(A)` este vectorul coloană format din elementele de pe diagonală principală a lui `A`. Pentru a produce o matrice diagonală având aceeași diagonală ca `A` se va utiliza `diag(diag(A))`. Analog cazului vectorial, `diag(A, k)` produce un vector coloană construit din a `k`-a diagonală a lui `A`. Astfel dacă

```
A = [2, 3, 5; 7, 11, 13; 17, 19, 23];
```

atunci

diag(A)

```
ans =  
     2  
    11  
    23
```

diag(A,-1)

```
ans =  
     7  
    19
```

`tril(A)` obține partea triunghiulară inferior a lui A (elementele situate pe diagonala principală și dedesubtul ei și în rest zero). Analog lucrează `triu(A)` pentru partea triunghiulară superior. Mai general, `tril(A,k)` dă elementele situate pe diagonala a k -a a lui A și dedesubtul ei, în timp ce `triu(A,k)` dă elementele situate pe a k -a diagonală a lui A și deasupra ei. Pentru A ca mai sus:

tril(A)

```
ans =  
     2     0     0  
     7    11     0  
    17    19    23
```



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

```
ans =
```

```
    0     3     5
    0     0    13
    0     0     0
```

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

```
ans =
```

```
    2     3     5
    7    11    13
    0    19    23
```

MATLAB posedă un set de funcții pentru generarea unor matrice speciale. Aceste matrice au proprietăți interesante care le fac utile pentru construirea de exemple și testarea algoritmilor. Ele sunt date în tabela 1. Funcția `gallery` asigură accesul la o colecție bogată de matrice de test creată de Nicholas J. Higham. Pentru detalii vezi `help gallery`.

<code>compan</code>	matrice companion
<code>gallery</code>	colecție de matrice de test
<code>hadamard</code>	matrice Hadamard
<code>hankel</code>	matrice Hankel
<code>hilb</code>	matrice Hilbert

invhilb	inversa matricei Hilbert
magic	pătrat magic
pascal	matricea Pascal (coeficienți binomiali)
rosser	matrice simetrică pentru testarea valorilor proprii
toeplitz	matrice Toeplitz
vander	matrice Vandermonde
wilkinson	matricea lui Wilkinson pentru testarea valorilor proprii

Tabela 1

Indexarea și notația ":"

Pentru i și j scalari $i:j$ desemnează vectorul $[i, i+1, \dots, j]$ (pasul este 1)

Pentru un pas diferit, s , se folosește $i:s:j$.

1:5

```
ans =  
     1     2     3     4     5
```

4:-1:-2

```
ans =  
     4     3     2     1     0    -1    -2
```

0:.75:3

```
ans =  
     0    0.7500    1.5000    2.2500    3.0000
```

Elementele individuale ale unei matrice se accesează prin $A(i, j)$, $i \geq 1$, $j \geq 1$. Indicii zero sau negativi nu sunt admiși în MATLAB.

$A(p:q, r:s)$ desemnează submatricea obținută din intersecția liniilor de la p la q și coloanelor de la r la s .

$A(i, :)$ linia i , $A(:, j)$ coloana j

$A(v, w)$ unde v și w sunt vectori selectează submatricea cu liniile date de v și coloanele date de w

$A(:)$ desemnează matricea A privita ca vector coloana (coloană după coloană).

Dacă $A(:)$ apare în stânga se completează A păstrându-i forma.

```
A=zeros(3); A(:)=primes(23); A=A'
```

$A =$

```

      2      3      5
      7     11     13
     17     19     23
```

$\text{linspace}(a, b, n)$ generează n puncte echidistante în intervalul $[a, b]$. Implicit $n=100$.

echivalent $a : (b-a) / (n-1) : b$.

$[]$ desemnează matricea vidă; utilă la ștergeri și indicator de poziție într-o listă de argumente.

```
A(2,:)=[]
```

$A =$

```

      2      3      5
     17     19     23
```

Operații în sens matricial și în sens tablou

Operațiile asupra matricelor se pot realiza în două moduri: în sens matricial, după regulile algebrei matriciale și în sens tablou, adică element cu element.

Operația	Sens matricial	Sens tablou
Adunare	+	+
Scadere	-	-
Inmulțire	*	.*
Împărțire	/	./
Împărțire stângă	\	.\
Ridicare la putere	^	.^

A/B este soluția ecuației matriciale $X*B=A$. $A\backslash B$ este soluția ecuației matriciale $A*X=B$.

A=[1 2; 3 4], B=ones(2)

A =

```
1    2
3    4
```

B =

```
1    1
1    1
```

A+B

```
ans =  
      2      3  
      4      5
```

A*B

```
ans =  
      3      3  
      7      7
```

A\B

```
ans =  
     -1     -1  
      1      1
```

A.*B, B./A

```
ans =  
      1      2  
      3      4
```

```
ans =  
  1.0000    0.5000  
  0.3333    0.2500
```

A^2

```
ans =  
     7     10  
    15     22
```

A.^2

```
ans =  
     1     4  
     9    16
```

Operația \wedge permite ca exponentul să fie un tablou când dimensiunile bazei și exponentului coincid, sau când baza este un scalar:

x=[1 2 3]; y=[2 3 4]; z=[1 2; 3 4];

x.^y

```
ans =  
     1     8    81
```

2.^x

```
ans =  
     2     4     8
```

2.^z

```
ans =
     2     4
     8    16
```

Calculul inversei se poate face cu `inv` iar al determinantului cu `det`. Dacă exponentul este negativ, ridicarea la A^n înseamnă $\text{inv}(A)^{-n}$

Transpusa conjugată a matricei A se obține cu A' . Dacă A este reală, atunci aceasta este transpusa obișnuită. Transpusa fără conjugare se obține cu $A.'$. Alternative funcționale: `ctranspose(A)` și `transpose(A)`

Produsul scalar a doi vectori `dot(x, y)`, sau dacă x și y sunt vectori coloană $x' * y$.

Produsul vectorial `cross(x, y)`

Exemple:

```
x=[-1 0 1]'; y=[3 4 5]';
```

```
x'*y
```

```
ans =
     2
dot(x, y)
```

```
ans =
     2
cross(x, y)
```



```
ans =  
-4  
8  
-4
```

Expandarea scalarilor: dacă un scalar se aduna la o matrice sau dacă se înmulțește (împarte) o matrice cu un scalar, scalarul se operează cu fiecare element al matricei:

```
[4,3;2,1]+4
```

```
ans =  
8      7  
6      5
```

```
[3 4 5; 4 5 6]/12
```

```
ans =  
0.2500    0.3333    0.4167  
0.3333    0.4167    0.5000
```

Funcțiile elementare se aplică matricelor și vectorilor punctual (element cu element):

```
sin(A)
```

```
ans =  
0.8415    0.9093  
0.1411   -0.7568
```

Funcțiile de matrice în sensul algebrei liniare au numele terminat în m: expm, logm, sqrtm, etc.

De exemplu pentru

A = [2 2; 0 2]

A =

2	2
0	2

avem

sqrt(A)

ans =

1.4142	1.4142
0	1.4142

sqrtm(A)

ans =

1.4142	0.7071
0	1.4142

ans*ans

ans =

2.0000	2.0000
0	2.0000

Analiza datelor

max	Maximul
min	Minimul
mean	Media
median	Mediana
std	Abaterea medie pătratică
var	Dispersia
sort	Sortare în ordine crescătoare
sum	Suma elementelor
prod	Produsul elementelor
cumsum	Suma cumulată
cumprod	Produsul cumulat
diff	Diferența elementelor

Cel mai simplu mod de aplicare – asupra unui vector

```
x=[4 -8 -2 1 0]; [min(x) , max(x) ]
```

```
ans =  
    -8     4
```

Aplicate asupra matricelor acționează pe coloană.

```
A = [0, -1, 2; 1, 2, -4; 5 -3 -4]
```

```
A =
```

```
    0    -1     2  
    1     2    -4  
    5    -3    -4
```

```
max(A)
```

```
ans =
```

```
    5     2     2
```

```
[m,i]=min(A)
```

```
m =
```

```
    0    -3    -4
```

```
i =
```

```
    1     3     2
```

```
min(min(A))
```

```
ans =
```

```
   -4
```

```
min(A(:))
```

```
ans =
```

-4

Funcțiile `max` și `min` pot acționa și pe linii printr-un al treilea argument.

```
max (A, [], 2)
```

```
ans =
```

2

2

5

Funcțiile `sum` și `sort` pot fi facute sa acționeze pe linii printr-un al doilea argument. Vezi `help sort` sau `doc sort`.

```
x=(1:8).^2
```

```
x =
```

1

4

9

16

25

36

49

64

```
diff(x)
```

```
ans =
```

3

5

7

9

11

13

15

Operatori relaționali și logici

`==`, `~=`, `<`, `>`, `<=`, `>=`

Comparația între scalari produce 1 dacă relația este adevărată și 0 în caz contrar. La comparația între matrice de aceeași dimensiune sau între matrice și scalari rezultatul este o matrice de 0 și 1.

Comparația se face element cu element. Exemple:

```
A = [1, 2; 3, 4]; B = 2*ones(2);
```

```
A == B
```

```
ans =
```

```
    0    1  
    0    0
```

```
A > B
```

```
ans =
```

```
    0    0  
    1    1
```

`isequal` testează dacă două matrice sunt identice

```
isequal(A,B)
```

```
ans =
```

```
    0
```

ischar	Testează dacă argumentul este șir de caractere (string)
isempty	Testează dacă argumentul este vid
isequal	Testează dacă tablourile sunt identice
isfinite	Testează dacă elementele unui tablou sunt finite
isieee	Testează dacă mașina utilizează aritmetica IEEE
isinf	Testează dacă elementele unui tablou sunt inf
islogical	Testează dacă argumentul este un tablou logic
isnan	Test de NaN
isnumeric	Testează dacă argumentul este numeric
Isreal	Testează dacă argumentul este tablou real
issparse	Testează dacă argumentul este tablou rar

operatori logici &, |, ~, xor, all, any. Exemple:

```
x = [-1 1 1]; y = [1 2 -3];
```

```
x>0 & y>0
```

```
ans =
```

```
0      1      0
```

```
x>0 | y>0
```

```
ans =
```

```
1      1      1
```

```
xor(x>0,y>0)
```

```
ans =  
    1    0    1  
any(x>0)
```

```
ans =  
    1  
all(x>0)
```

```
ans =  
    0
```


Precedența operatorilor

1. transpose (.'), power (.^), complex conjugate transpose ('), matrix power (^)
2. unary plus (+), unary minus (-), logical negation (~)
3. multiplication (.*), right division (./), left division (.\), matrix multiplication (*), matrix right division (/), matrix left division (\)
4. addition (+), subtraction (-)
5. colon operator (:)
6. less than (<), less than or equal to (<=), greater than (>), greater than or equal to (>=), equal to (==), not equal to (~=)
7. element-wise logical AND (&)
8. element-wise logical OR (|)
9. short-circuit logical AND (&&)
10. short-circuit logical OR (||)

`all` și `any` aplicate matricelor lucrează pe coloane, rezultatul fiind un vector linie.
`all (all (A==B))` este echivalent cu `isequal (A, B)`

`find` returnează indicii corespunzători elementelor nenule ale unui vector

```
x = [-3 1 0 -inf 0];
```

```
f = find(x);
```

Rezultatul lui `find` poate fi utilizat apoi la selecții

```
x(f)
```

```
ans =  
    -3     1   -Inf
```

Selectăm elementele finite ale lui x

```
x(find(isfinite(x)))
```

```
ans =  
    -3     1     0     0
```

înlocuim elementele negative cu 0

```
x(find(x<0))=0
```

```
x =  
     0     1     0     0     0
```

find aplicat unei matrice acționează ca și cum matricea ar fi transformată într-un vector coloană.

```
A = [4 2 16; 12 4 3], B = [12 3 1; 10 -1 7]
```

```
A =  
     4     2    16  
    12     4     3
```

```
B =  
    12     3     1  
    10    -1     7
```

```
f = find(A<B)
```

```
f =
```

```
1  
3  
6
```

```
A(f)=0
```

```
A =
```

```
0    0   16  
12    4    0
```

Pentru matrice, `find` se poate utiliza și sub forma

```
[i,j] = find(A)
```

```
i =
```

```
2  
2  
1
```

```
j =
```

```
1  
2  
3
```

Rezultatele operatorilor logici și a funcțiilor logice sunt exemple de tablouri logice. Tablourile logice se pot crea și prin aplicarea funcției `logical` unui tablou numeric. Tablourile logice pot fi utilizate la indexare.

```

clear; y = [1 2 0 -3 0]
y =
    1    2    0   -3    0
i1 = logical(y)

i1 =
    1    1    0    1    0
i2 = ( y~=0 )

i2 =
    1    1    0    1    0
i3 = [1 1 0 1 0]

i3 =
    1    1    0    1    0
whos

```

Name	Size	Bytes	Class	Attributes
i1	1x5	5	logical	
i2	1x5	5	logical	
i3	1x5	40	double	
y	1x5	40	double	

```
y(i1),y(i2)
```

```
ans =
```

```
1      2     -3
```

```
ans =
```

```
1      2     -3
```

```
isequal(i2,i3)
```

```
ans =
```

```
1
```

```
y(i3)
```

??? Subscript indices must either be real positive integers or logicals.

A(M) unde M este un tablou logic de aceeași dimensiune ca și A, extrage elementele lui A corespunzând elementelor lui M cu partea reală nenulă. Chiar dacă i2 are aceleași elemente ca i3 (și la comparație ele ies egale), doar tabloul logic i2 poate fi utilizat la indexare.

Un apel la find poate fi uneori evitat dacă argumentul său este un tablou logic. În exemplul precedent, x(find(isfinite(x))) poate fi înlocuit cu x(isfinite(x)). Se recomandă utilizarea lui find pentru claritate.