Alkalmazott matematika

Baran Ágnes

Octave/Matlab alapok

Vektorok Octave/Matlab-ban

Megkülönbözteti a sor- és oszlopvektorokat

Sorvektorok

Az $a = (-1.2, 3.1, 4.7, 1.9)$ vektor megadása elemei felsorolásával:

- az elemeket vesszővel választjuk el:

  ```
  a =[-1.2, 3.1, 4.7, 1.9]
  ```

- vagy az elemeket szóközzel választjuk el:

  ```
  a=[-1.2 3.1 4.7 1.9]
  ```

A vektorkoordináták számozása 1-gyel kezdődik, `a(i)` az *a* vektor *i*-edik koordinátája.

`length(a)` az *a* vektor koordinátáinak száma

`a=[]` üres vektor

Vektorok, mint szabályos sorozatok

A kettőspont operátorral

- a $b = (1, 2, 3, 4, 5)$ vektor:

  ```
  b = 1:5
  ```

- a $c = (5, 4, 3, 2, 1)$ vektor:

  ```
  c = 5:-1:1
  ```

- a $d = (2, 2.2, 2.4, 2.6, 2.8, 3)$ vektor

  ```
  d=2:0.2:3
  ```

Általában:

```
x=elsoelem:lepeskoz:utolsoelem
```

ahol a lépésköz negatív is lehet, vagy

```
x=elsoelem:utolsoelem
```

ekkor a lépésköz 1.

Vektorok, mint szabályos sorozatok

A `linspace` függvénnyel:

- az $e = (1, 1.2, 1.4, 1.6, 1.8, 2)$ vektor

  ```
  e=linspace(1,2,6)
  ```

- egy 100 koordinátából álló f vektor

  ```
  f=linspace(1,2)
  ```

Általában:

```
x=linspace(elsoelem,utolsoelem,elemekszama)
```

ahol a koordináták egyforma lépésközzel követik egymást, vagy

```
x=linspace(elsoelem,utolsoelem)
```

ekkor a koordináták száma 100.

Oszlopvektorok

Oszlopvektorok megadása

- elemeinek felsorolásával (a vektor koordinátáit pontosvesszővel választjuk el)

  `m=[-3;0;7]`

- egy sorvektor transzponálásával: `n=[1 -2 4 -1]'`
  (**valójában a ' jel konjugált transzponáltat eredményez, a konjugálás nélküli transzponálás:** `a.'` **vagy** `transpose(a)`)

`x(i)` és `length(x)` az x vektor i-edik koordinátája és az x vektor koordinátáinak száma (ugyanúgy mint a sorvektoroknál)

`size(x)` az x vektor mérete (sorvektoroknál az `[1 length(x)]` vektor, oszlopvektoroknál a `[length(x) 1]` vektor)

Vektorok konstruálása más vektorokból

- `[a b]` két sorvektor egymás után fűzése
- `[m;n]` két oszlopvektor egymás után fűzése
- `[-4 a 3 -1]` sorvektor bővítése újabb elemekkel
- `[1;m;-3]` oszlopvektor bővítése újabb elemekkel
- `h(2:4)` a *h* vektor 2., 3. és 4. koordinátájából álló vektor
- `h([1 4 5])` a *h* vektor 1., 4. és 5. koordinátájából álló vektor
- `h(2)=[]` elhagyja a *h* vektor 2. koordinátáját
- `h([2 4])=[]` elhagyja a *h* vektor 2. és 4. koordinátáját

Fontos! Ha `a=[-1 3 2]` akkor az `a(6)=4` utasítás eredménye az `a=[-1 3 2 0 0 4]` vektor (a legkisebb olyan vektor, amelyben van értelme a `a(6)=4` utasításnak, a nemdefiniált elemeket 0-kal tölti fel. **Megváltozik a vektor mérete, erre nem figyelmeztet!**)

Néhány hasznos függvény

- `min(x)` és `max(x)` az x vektor legkisebb és legnagyobb eleme
- `sort(x)` az x elemeit növekvő sorrendbe rendezi
- `sort(x,'descend')` az x elemeit csökkenő sorrendbe rendezi
- `flip(x)` az x elemeit fordított sorrendben sorolja fel
- `length(x)` az x vektor elemeinek a száma
- `sum(x)` az x vektor elemeinek összege
- `prod(x)` az x vektor elemeinek szorzata
- `mean(x)` az x vektor elemeinek átlaga
- `x(3)` az x vektor harmadik eleme
- `x(1:3)` az x vektor első három eleme
- `x(3:end)` az x vektor minden elemei a harmadiktól az utolsóig

Műveletek vektorokkal

Ha `a` és `b` két ugyanolyan méretű vektor, akkor

- `a+b` ill. `a-b` a két vektor összege és különbsége
- `x=a+1` egy ugyanolyan méretű vektor, mint `a`, $x_i = a_i + 1$
- `x=a.^2` egy ugyanolyan méretű vektor, mint `a`, $x_i = a_i^2$.
- `x=a.*b` egy ugyanolyan méretű vektor, mint `a` és `b`, $x_i = a_i b_i$
- `x=a./b` egy ugyanolyan méretű vektor, mint `a` és `b`, $x_i = \frac{a_i}{b_i}$
- `x=1./a` egy ugyanolyan méretű vektor, mint `a`, $x_i = \frac{1}{a_i}$
- `dot(a,b)` az `a` és `b` skaláris szorzata

Fontos! A műveleti jel előtti pont a művelet elemenkénti végrehajtását eredményezi

`sin`, `cos`, `tan`, `exp`, `log`, `sqrt`, `abs`, stb. mind elemenként hajtódik végre.

`NaN` : Not a Number (pl. 0/0, `Inf/Inf`)

1. feladat

(a) Az elemek egyenkénti begépelése nélkül állítsa elő az alábbi vektorokat!

   (1) $a = (0, 1, \cdots, 30)$
   (2) $b = (2, 4, 6, \ldots, 100)$,
   (3) $c = (2, 1.9, 1.8, \cdots, 0.1, 0)$
   (4) $d = (0, 3, 6, \ldots, 27, 30, -100, 30, 27, \cdots, 6, 3, 0)$
   (5) $e = \left(\frac{1}{2}, \frac{1}{3}, \cdots, \frac{1}{20}\right)$
   (6) $f = \left(\frac{1}{2}, \frac{2}{3}, \cdots, \frac{19}{20}\right)$

(b) Legyen x egy adott 100 elemű sorvektor. Az x vektorból állítsa elő azt az y vektort, melynek elemei

   (1) az x vektor elemei fordított sorrendben felsorolva,
   (2) az x vektor első 5 eleme,
   (3) az x vektor elemei ugyanolyan sorrendben, kivéve az x 4. elemét
   (4) az x vektor elemei ugyanolyan sorrendben, kivéve az x 5., 72. és 93. elemét
   (5) az x vektor páratlan sorszámú elemei
   (6) az x vektor 2., 5., 17. és 81. eleme.

2. feladat

Legyen x egy adott sorvektor. A `for` utasítás használata nélkül az x vektorból állítsa elő azt az y vektort, melynek i-edik eleme

(1) $x(i) + 2$

(2) $x(i)^2$

(3) $1/x(i)$

(4) $\sin(x(i)^3 - 1)$

(5) $x(i) - i$

Mátrixok Octave/Matlab-ban

Mátrix megadása elemenként

$A = [1, 2, 3; 4, 5, 6; 7, 8, 9]$ vagy $A = [1 \quad 2 \quad 3; \quad 4 \quad 5 \quad 6; \quad 7 \quad 8 \quad 9]$ eredménye:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

(Az egy sorban álló elemeket vesszővel vagy szóközzel, a sorokat pontosvesszővel választjuk el.)

A mátrixelemek számozása $(1, 1)$-gyel kezdődik.

`A(i,j)` a mátrix (i, j)-edik eleme.

Mátrixok megadása

Vektorok összefűzésével

Ha `a=[1 -2 0]`; `b=[2 -11 7]`; `m=[-3;0;7]`; `n=[1; -2; 0]`; akkor `B=[a;b]` eredménye:

$$B = \begin{pmatrix} 1 & -2 & 0 \\ 2 & -11 & 7 \end{pmatrix}$$

`C=[a'  b']` és `D=[m  n]` eredménye:

$$C = \begin{pmatrix} 1 & 2 \\ -2 & -11 \\ 0 & 7 \end{pmatrix} \qquad D = \begin{pmatrix} -3 & 1 \\ 0 & -2 \\ 7 & 0 \end{pmatrix}$$

Mátrixok bővítése

Az előbb definiált mátrixokkal, vektorokkal:
`E=[A;a]` vagy `E=[A;[1,-2,0]]` eredménye

$$E = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 1 & -2 & 0 \end{pmatrix}$$

Tehát: *[mátrix „sortörés" (azaz ;) sorvektor]*

Az `F=[A m]` vagy `F=[A, m]` eredménye

$$F = \begin{pmatrix} 1 & 2 & 3 & -3 \\ 4 & 5 & 6 & 0 \\ 7 & 8 & 9 & 7 \end{pmatrix}$$

Tehát: *[mátrix szóköz vagy vessző oszlopvektor]*

Mátrixok bővítése

`G=[C D]` és `H=[C;D]` eredménye

$$G = \begin{pmatrix} 1 & 2 & -3 & 1 \\ -2 & -11 & 0 & -2 \\ 0 & 7 & 7 & 0 \end{pmatrix} \qquad H = \begin{pmatrix} 1 & 2 \\ -2 & -11 \\ 0 & 7 \\ -3 & 1 \\ 0 & -2 \\ 7 & 0 \end{pmatrix}$$

`C(4,5)=9` eredménye:

$$C = \begin{pmatrix} 1 & 2 & 0 & 0 & 0 \\ -2 & -11 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 9 \end{pmatrix}$$

Megváltozik a mátrix mérete, erre nem figyelmeztet!

Hivatkozás elemekre, sorokra, oszlopokra, részmátrixokra

- `size(A)` az A mátrix mérete (egy kételemű sorvektor)
- `length(A)` egy skalár: `max(size(A))`
- `A(i,j)` az A mátrix (i,j)-edik eleme
- `A(i,:)` egy sorvektor, az A mátrix i-edik sora
- `A(:,j)` egy oszlopvektor, az A mátrix j-edik oszlopa
- `A(2:3,:)` az A mátrix 2. és 3. sora
- `A([1 2 4],:)` az A mátrix 1., 2. és 4. sora
- `A(:,[1 3])` az A mátrix 1. és 3. oszlopa
- `A(2:3,[1 3])` az A mátrix 2. és 3. sorának és 1. és 3. oszlopának metszetéből álló mátrix

Mátrixok „átszabása”

Sorok, oszlopok elhagyása mátrixokból

- `A(i,:)=[]` az *i*-edik sor elhagyása
- `A(:,j)=[]` a *j*-edik oszlop elhagyása
- `A([1 3],:)=[]` az 1. és 3. sor elhagyása
- `A(:,[1 3])=[]` az 1. és 3. oszlop elhagyása

Sor- és oszlopcsere

Az *i*-edik és *j*-edik sor illeve oszlop cseréje:
`A([i,j],:)=A([j,i],:)`, ill. `A(:,[i,j])=A(:,[j,i])`

Mátrixból vektor

`A(:)` az A mátrix elemei oszlopfolytonosan felsorolva

Néhány beépített mátrix

eye(n)	az $n \times n$-es egységmátrix
eye(n,m)	az $n \times m$-es egységmátrix
ones(n)	a csupa 1-esből álló $n \times n$-es mátrix
ones(n,m)	a csupa 1-esből álló $n \times m$-es mátrix
zeros(n)	a csupa 0-ból álló $n \times n$-es mátrix
zeros(n,m)	a csupa 0-ból álló $n \times m$-es mátrix

Néhány hasznos függvény

- `numel(A)` az A elemeinek száma
- `size(A)` az A mérete
- `length(A)` egyenlő `max(size(A))` értékével

Műveletek vektorok és mátrixok között

Legyen A és B két mátrix (melyek akár vektorok is lehetnek), c egy skalár. Az

$$\texttt{A+B}, \quad \texttt{A-B}, \quad \texttt{c*A}, \quad \texttt{A*B}, \quad \texttt{A\^{}2}$$

műveletek a hagyományos, lineáris algebrában értelmezett műveletek, feltéve, hogy A és B mérete megfelelő. Az

$$A + c$$

művelet eredménye: az A minden eleméhez hozzáadunk c-t. Az

$$A/B \quad \text{és} \quad A\backslash B$$

műveletek eredménye $A \cdot B^{-1}$ és $A^{-1} \cdot B$.

Műveletek vektorok és mátrixok között

Elemenkénti művelet

A műveleti jel előtti . jel a művelet elemenkénti végrehajtását eredményezi:

Az `A.*B` mátrix *ij*-edik eleme $a_{ij} * b_{ij}$,
az `A.^ 2` mátrix *ij*-edik eleme a_{ij}^2,
az `A./B` mátrix *ij*-edik eleme a_{ij}/b_{ij}.

A beépített függvények általában hívhatók mátrix argumentummal is, pl. `sin(A)`, `log(A)`, `exp(A)`, `abs(A)`, stb. Ilyenkor a függvény a mátrix minden elemére végrehajtódik.

3. feladat

Legyen $x = [-1 \quad 4 \quad 0]$, $y = [3 \quad -2 \quad 5]$
és $A = [-3 \quad 1 \quad -4; 6 \quad 2 \quad -5]$. Döntse el, hogy az alábbi utasítások közül melyik végrehajtható. Ha nem végrehajtható, akkor magyarázza meg miért, ha végrehajtható, akkor fogalmazza meg mi lesz az eredmény!

(1) $z = [x, y]$
(2) $z = [x; y]$
(3) $z = [x', y']$
(4) $z = [x'; y']$
(5) $z = [A; x]$
(6) $z = [A, x]$
(7) $z = [x; A; y]$
(8) $z = [A'; x]$
(9) $z = [A', x]$
(10) $z = [A', x']$
(11) $x + y$
(12) $x + y'$
(13) $A + y$
(14) $A + 2$
(15) $x./y$
(16) $A\,\hat{}\,2$
(17) $A.\hat{}\,2$

4. feladat

Legyen

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix}$$

Konstruálja meg (az elemek felsorolása nélkül) azt a B mátrixot, melyet úgy kapunk, hogy

(1) elhagyjuk az A mátrix első sorát,

(2) elhagyjuk az A mátrix 2. és 4. oszlopát,

(3) elhagyjuk az A mátrix utolsó sorát és oszlopát

(4) kétszer egymás mellé írjuk az A mátrixot,

(5) transzponáljuk az A mátrixot,

(6) felcseréljük az A mátrix 2. és 4. oszlopát

(7) négyzetre emeljük az A elemeit

(8) az A minden elemét megnöveljük 3-mal

(9) A minden elemének vesszük a négyzetgyökét

(10) A minden elemének vesszük a szinuszát

(11) az A első sorának második elemét kicseréljük -2-re

(12) az A 2. sorát kicseréljük a $[-1 \quad 0 \quad -2 \quad 3]$ vektorra

5. feladat

- Egy rövid utasítás segítségével állítsa elő az alábbi mátrixot!

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 20 & 18 & 16 & 14 & 12 & 10 & 8 & 6 \\ 2 & 4 & 8 & 16 & 32 & 64 & 128 & 256 \end{pmatrix}$$

- Az előző feladat A mátrixával vizsgálja meg az alábbi utasítások eredményét!

(1) `sum(A)`
(2) `sum(A,2)`
(3) `reshape(A,6,4)`
(4) `max(A)`
(5) `max(A,[],2)`
(6) `max(A,2)`
(7) `flipud(A)`
(8) `fliplr(A)`
(9) `size(A)`
(10) `length(A)`

Néhány, lineáris algebrában hasznos függvény

- `det(A)` az A determinánsa
- `inv(A)` az A inverze
- `dot(a,b)` az a és b vektorok belsőszorzata
- `norm(A)` az A 2-normája (mátrixok és vektorok esetén is)
- `norm(A,inf)` az A ∞-normája (mátrixok és vektorok esetén is)
- `norm(A,1)` az A 1-normája (mátrixok és vektorok esetén is)

Az $Ax = b$ lineáris egyenletrendszer megoldása (részletesen ld. később):

```
x=A\b
```

Néhány hasznos függvény

`diag`

- `diag(a)`
  ahol *a* egy vektor, egy négyzetes mátrixszal tér vissza, főátlójában az *a* vektorral
- `diag(a,k)`
  ahol *a* egy vektor, *k* egy egész, egy olyan mátrixszal tér vissza, aminek a *k*-adik átlója az *a* vektor. A 0. átló a főátló, onnan felfelé egyesével nő, lefelé egyesével csökken az átlók sorszáma.
- `diag(A)`
  ahol *A* egy mátrix (nem feltétlenül négyzetes) egy oszlopvektorral tér vissza, az *A* főátlóbeli elemeivel
- `diag(A,k)`
  ahol *A* egy mátrix, *k* egy egész, egy oszlopvektorral tér vissza, az *A* mátrix *k*-adik átlójának elemeivel.

Néhány hasznos függvény

`tril` és `triu`

- `tril(A)`
  Az A mátrix alsóháromszög részével tér vissza (a főátló és az alatta álló elemek, a többi 0)
- `triu(A)`
  Az A mátrix felsőháromszög részével tér vissza (a főátló és a felette álló elemek, a többi 0)
- `tril(A,k)`
  Az A mátrix k-adik átlójában és az alatta álló elemekkel tér vissza (a többi 0)
- `triu(A,k)`
  Az A mátrix k-adik átlójában és a felette álló elemekkel tér vissza (a többi 0)

Függvények írása

Az Octave/Matlab függvények szerkezete:

```
function kimenovaltozok=fvneve(bemenovaltozok)
 utasitasok
end
```

Fontos! A fenti függvényt fvneve.m néven kell elmenteni.

Példák.

```
function  y=masodf(x)
   y=2*x.^2-3*x+5;
end
```

Ekkor a `y=masodf(x)` utasítás eredménye a $2x^2 - 3x + 5$ kifejezés értéke, ahol x akár vektor is lehet, ebben az esetben a függvény elemenként hajtódik végre és y is vektor (ezt az teszi lehetővé, hogy a fv-ben minden művelet végrehajtható vektorokra is, mivel a négyzetreemelés jele elé kitettük a . jelet)

Logikai kifejezések

- $<$, $<=$, $>$, $>=$, $==$, $\sim=$ (két mátrixra is alkalmazhatóak, ilyenkor elemenként történik az összehasonlítás)
- `A&B`, `A|B`, `~A`, `xor(A,B)` (az kiértékelés elemenként)

Logikai függvények

- `all(A)`
  egy sorvektorral tér vissza, oszloponként megvizsgálja, hogy minden elem 0-tól különböző-e (ugyanaz, mint `all(A,1)`)
- `all(A,2)`
  egy oszlopvektorral tér vissza, soronként megvizsgálja, hogy minden elem 0-tól különböző-e
- `any(A)`
  egy sorvektorral tér vissza, oszloponként megvizsgálja, hogy van-e 0-tól különböző elem (ugyanaz, mint `any(A,1)`)
- `any(A,2)`
  egy oszlopvektorral tér vissza, soronként megvizsgálja, hogy van-e 0-tól különböző elem

Logikai függvények

- `ind=find(A)`
  a nemnulla elemek sorszámaival tér vissza, ahol az elemek számozása oszlopfolytonosan történik. Ha A egy sorvektor, akkor a visszaadott érték is az, egyébként oszlopvektor
- `ind=find(A,n)`
  az A első n darab nemnulla elemének sorszámával tér vissza
- `ind=find(A,n,last)`
  az A utolsó n darab nemnulla elemének sorszámával tér vissza
- `[rowI,colI]=find(A)`
  a nemnulla elemek sor- és oszlopindexeivel tér vissza
- `[rowI,colI,elem]=find(A)`
  a nemnulla elemek sor- és oszlopindexeivel, illetve a nemnulla elemekkel tér vissza

Logikai függvények

A `find` függvény argumentumába A helyett bármilyen logikai kifejezés is beírható, pl:

- `find(A<=B)`
- `find(A==5)`
- `find(A>5,4)`
- `find(A>5,4,last)`
- `find(A<5 & A>4)`
- `find(abs(A-2)<=0.01)`

`logical(A)`
egy logikai tömbbel tér vissza: az A nemnulla elemeinek a logikai 1, a nulla elemeknek a logikai 0 felel meg

Elágazások, if-else

```
if logikai kifejezés
  utasítások
elseif logikai kifejezés
  utasítások
else
  utasítások
end
```

Példa

```
N=input('Adjon meg egy egesz szamot: ');
if mod(N,3)==0
   disp('Oszthato 3-mal');
elseif mod(N,3)==1
   disp('3-mal osztva 1 maradekot ad');
else
   disp('3-mal osztva 2 maradekot ad');
end
```

Vezérlő utasítások

- `break`
  kilép az aktuális for- vagy while-ciklusból (csak a legbelsőből)
- `continue`
  a for- vagy while-ciklus következő lépésével folytatja
- `pause`
  billentyűlenyomásig felfüggeszti a program futását
- `pause(n)`
  n másodpercre felfüggeszti a program futását
- `return` leállítja az m-fájl futását
- `error('uzenet')` az m-fájl futása `uzenet` hibaüzenettel befejeződik

6. feladat

Legyen $x = [0 \quad -1 \quad 2 \quad 0 \quad 4 \quad 4]$ és $y = [-1 \quad -2 \quad 3 \quad 1 \quad 0 \quad 4]$.
Vizsgálja meg az alábbi logikai kifejezések értékét!

(1) $x == y$
(2) $x <= y$
(3) $x > y$
(4) $x > 0$
(5) $y <= 3$
(6) $x|y$
(7) $x\&y$

7. feladat

Az előző feladat vektoraival vizsgálja meg az alábbi kifejezések értékét

(1) $\texttt{find}(x == y)$
(2) $\texttt{find}(x <= y)$

8. feladat

Legyen `a=rand(1,20)` Készítse el azt a *b* vektort, amely az *a* 0.5-nél nagyobb elemeit tartalmazza!

9. feladat

Írjon egy-egy Matlab függvényt, amely tetszőleges n esetén előállítja az alábbi mátrixokat $n \times n$-es méretben!

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 3 & 0 & 3 & 0 & 0 \\ 0 & 6 & 0 & 4 & 0 \\ 0 & 0 & 9 & 0 & 5 \end{pmatrix}, \quad B = \begin{pmatrix} -2 & 2 & 2 & 2 & 2 \\ 0 & -2 & 2 & 2 & 2 \\ 0 & 0 & -2 & 2 & 2 \\ 0 & 0 & 0 & -2 & 2 \\ 0 & 0 & 0 & 0 & -2 \end{pmatrix}$$

10. feladat

Írjon egy függvényt, mely adott A mátrix esetén előállítja azt a B mátrixot, amelyet úgy kapunk, hogy az A minden sorának végére odaírjuk az abban a sorban álló elemek átlagát!

11. feladat

Írjon két függvényt, melyek adott x vektor esetén kiszámítják azt az y vektort, melynek minden koordinátája 1-gyel nagyobb az x vektor megfelelő koordinátájánál! Az első függvényben `for`-ciklussal koordinátánként végezze a műveletet, a másodikban használja a vektorizált alakot (`y=x+1`). Mérje le a futási időket akkor, amikor a függvényeket egy 1000000 elemű véletlen vektorral hívja! (Használja a `tic` és `toc` függvényeket.) Vizsgálja meg azt is, hogy mit jelent futási időben, ha az első esetben nem inicializálja az y vektort.