

3. GYAKORLAT

MÁTRIXARITMETIKA

LINEÁRIS ALGEBRAI MŰVELETEK

Az összeadás és kivonás művelettel – egyszerűségük miatt – csak egy példában foglalkozunk.

Feladat

Legyen $A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$. Adjuk meg a B csupa egyes mátrixot (a *ones* függvénnyel) úgy, hogy $A + B'$ elvégezhető legyen!

Szorzás művelet

Két mátrix csak akkor szorozható össze, ha az első oszlopainak száma a második tényező sorainak számával egyezik meg.

Eml.: a `size(A)` függvényhívás az A mátrix sorainak és oszlopainak számát adja vissza.

Például $A = \text{ones}(4, 5)$; `size(A)` eredménye `[4 5]`.

A `size(A, melyiket)` függvényhívás csak a lekérdezett dimenzió szerinti méretet adja vissza, azaz `size(A, 1) = 4` és `size(A, 2) = 5` lesz.

Így az A és B mátrixok összeszorozhatósága eldönthető a következő hasonlítással:

```
>> size(A,2) == size(B,1)      % belső indexek azonosak-e?
```

Ha ez teljesül, akkor a $C = A*B$ mátrix mérete

```
[size(A,1) size(B,2)]          % külső indexek
```

Feladat

Az $A = \text{ones}(3, 2)$ és a $B = \text{magic}(4)$ mátrixokat szeretnénk $A*B$ módon összeszorozni. Az A mátrix melyik méretét és mennyire kellene változtatni, hogy ez sikerülhessen?

Ha a szorzás mindkét irányban elvégezhető, általában akkor sem kommutatív művelet, azaz $A*B$ és $B*A$ többnyire nem azonosak.

Feladat

Legyen $A = \begin{bmatrix} 1 & 3 & 4 & 6 & 7 & 9 \end{bmatrix}$, $B = \text{ones}(3)$. Ellenőrizzük, hogy egyenlő-e az $A*B$ és $B*A$ szorzat!

Hajtsuk végre ugyanezt a

$P = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 1 & 3 & 6 & 10 \\ 1 & 4 & 10 & 20 \end{bmatrix}$ és az

$IP = \begin{bmatrix} 4 & -6 & 4 & -1 & -6 & 14 & -11 & 3 \\ 4 & -11 & 10 & -3 & -1 & 3 & -3 & 1 \end{bmatrix}$ mátrixokkal is.

Speciális mátrixokra a szorzatmátrix is lehet speciális tulajdonságú.

Feladat

Legyen $X = \text{randi}([1 \ 19], 3)$ % véletlen mátrix generálása: ([határok], méret)
 $SW12 = [0 \ 1 \ 0; 1 \ 0 \ 0; 0 \ 0 \ 1]$, $SW13 = [0 \ 0 \ 1; 0 \ 1 \ 0; 1 \ 0 \ 0]$, $SW23 = [1 \ 0 \ 0; 0 \ 0 \ 1; 0 \ 1 \ 0]$
 Az $SW12 \cdot X$ szorzat felcseréli az X mátrix 1. és 2. sorát. (*swap*: felcserélés)
 Nézzük meg a többi swap mátrixszal balról történő szorzást is!
 Mi lesz az SW mátrixok páros kitevős hatványa és mi lesz az SW mátrixok páratlan kitevős hatványa?
 Mi történik, ha az SW mátrixokkal jobbról szorozzuk az X mátrixot?

Osztás művelet

Matlab a mátrixszal való osztást valamilyen inverzzel való szorzással hajtja végre.
 Tudjuk, hogy valódi inverz (*inv* parancs) akkor létezik, ha a – négyzetes – mátrix determinánsa nem 0, ill. ha a mátrix rangja megegyezik a sorainak (oszlopainak) számával.

Feladat

a./ Legyen $E = [1 \ 2; 3 \ a]$. Válasszuk meg az a paraméter értékét úgy, hogy a mátrix rangja

- 1 legyen;
- 2 legyen!

Az utóbbi esetben határozzuk meg az inverzet, és szorzással ellenőrizzük, hogy valóban az inverz mátrixot kaptuk meg!
 Mit kapunk, ha az 1 rangú esetben próbálunk inverz mátrixot számoltatni az *inv* paranccsal?

b./ Mi az SW mátrixok inverze? Magyarázzuk meg az eredményt!

A valódi inverznél általánosabban értelmezett a bal- és a jobbinverz, ill. a pszeudoinverz.
 Előbbiek az egységmátrix felhasználásával határozhatók meg, utóbbi előállítására pedig a Matlab beépített parancsot biztosít.

Feladat

Előkészítésként nézzük meg a súgóban (*doc*) a \ és a / (*mldivide* és *mrdivide*) műveletek leírását!
 Legyen $A = [1 \ 2; 3 \ 4]$ és $B = [10 \ 7; 22 \ 15]$.
 Határozzuk meg azt az X mátrixot, amelyre $A \cdot X = B$.
 Határozzuk meg azt az Y mátrixot, amelyre $Y \cdot B = A$.
 Határozzuk meg azt az Z mátrixot, amelyre $Z \cdot A = B$.

Példa

Előkészítésként nézzük meg a súgóban a *pinv* parancs leírását!
 Generáljunk egy 4×4 -es mátrixot egyjegyű számokkal a *randi* függvény (lásd lent) segítségével.

```
>> P = randi([1 9], 4), rank(P)
```

Ha a rang 4 (általában ez teljesül), akkor először határozzuk meg az inverzet külön (*inv*(P)), és jelenítsük meg tört formátumban is (*rats* parancs).

Ezután ellenőrizzük, hogy az inverz különféle számításával kapott eredmények mennyire egyeznek a valódi inverzzel!

```
>> P^-1 - inv(P)           % -1. hatvány
>> P\eye(4) - inv(P)       % jobbinverz (P az egység. balosztója)
>> eye(4)/P - inv(P)       % balinverz (P az egység. jobbosztója)
>> pinv(P) - inv(P)        % pszedoinverz
```

(Ha az eredménymátrix minden eleme 0, akkor a két számítási mód teljesen azonos eredményt szolgáltatott.)

Feladat

Legyen $X = [8 \ 5 \ 2 \ 2; 6 \ 4 \ 1 \ 4; 3 \ 0 \ 1 \ 0]$.

Nem négyzetes mátrixnak nincs inverze, csak pszeudoinverze. Ellenőrizzük, hogy olyan esetben, mint most (az X mátrixnak kevesebb sora van, mint oszlopa) a pszeudoinverz egy jobbinverz, azaz $X \cdot \text{pinv}(X)$ lesz az egységmátrix. (Nézzük meg, hogy az $X \backslash \text{eye}(3)$ módon számolt jobbinverz ettől eltér-e!)

Határozzuk meg az X transzponáltjának a pszeudoinverzét is! Ez balinverz?

NEVEZETES MÁTRIXOK

Véletlen számok

`rand(méret)` – egyenletes eloszlású (0, 1) intervallumbeli valósak (álvéletlenek)

`randi(határok, méret)` – egyenletes eloszlású egészek

(A `randn` függvény használatát lásd a házi feladatok között.)

```
>> rand(5)                  % 5×5 méretű mátrix
>> rand(1,5), rand(5,1)    % sorvektor, oszlopvektor
>> A = ones(10,4); rand(size(A)) % 10×4-es véletlen mátrix
>> randi([1 90], size(1:5))
% egy lottóhúzás eredménye lehet, az ismétlődés esélye kicsi
```

Egységmátrix, csupa egyes és csupa nulla mátrix

Ezen nevezetes mátrixok könnyen megjegyezhetők, mert az angol nevük alapján a tartalmuk egyértelmű és a méretmegadás azonos konvenciót követ (`ones`, `zeros`, `eye`).

Paraméterezésük: méret, típus. Itt a típus elmaradhat, a méret egyetlen, vagy több numerikus adat, amelyeknek az egész része számít.

Példák

```
>> ones(3)                  % 3×3 méretű csupa egyesekből álló mátrix
>> ones(3, 4)               % 3×4 méretű csupa egyesekből álló mátrix
>> zeros(3, 'int8')         % 3×3 méretű 1 bájtton tárolt 0 adatok
```

```
>> eye(3, 5)           % 3x5-ös egységmátrix, csak a bal felső
saroktól induló átlóban vannak egyesek, a többi elem nulla
>> A = [1:3; 4:6; 7:9], E = eye(size(A)) % értelmezzük!
```

Feladat

Eml.: tudjuk, hogy a 0-val való osztás problémás művelet.
Mik lesznek az `eye(3)./zeros(3)` hányados elemei?

Hilbert- és Pascal-mátrix

`hilb(n)` – $n \times n$ méretű Hilbert-mátrix, amelynek elemei a természetes számok reciprokai a következő szabály szerint:

$$h(i, j) = 1/(i + j - 1)$$

A törtalakot (karakterláncként) a `rats(hilb(n))` szolgáltatja. A Hilbert-mátrixok determinánsai egyre kisebbek! (Gyorsan csökkenő sorozat...)

`invhilb(n)` – a Hilbert-mátrix inverze, amelynek elemei egyre nagyobb egészeket tartalmaznak.

```
>> H3 = hilb(3), d3 = det(H3), T3 = rats(H3)
```

Feladatok

Állítsuk elő az 5×5 -ös Hilbert-mátrixot! Mennyi a 3. oszlop elemeinek az összege?

Melyik Hilbert mátrixnak lesz először kisebb a determinánsa, mint az *eps* érték?

Állapítsuk meg az `hilb(4)`, `hilb(5)`, `hilb(6)` mátrixok inverzeinek legnagyobb elemét (*max* függvény)!

`pascal(n)` – a binomiális együtthatókból képezett mátrix, amelynek anti-átlóiban az $(a + b)$ egyre növekvő egészhatványainak együttható-sorozata van.

Feladatok

Állapítsuk meg az $(a + b)^7$ kifejtésében az a^5b^2 tag együtthatóját!

Melyik a Pascal-háromszög 8. sorának 2. legnagyobb eleme?

(Itt a hagyományos háromszöges elrendezést követjük, a háromszög csúcsában található egy darab 1-es elemet 0. sornak tekintjük.)

Feladat

Építsük fel az egységmátrix, a csupa nulla mátrix, a csupa egyes mátrix, a Pascal-mátrix, a Hilbert-mátrix és egyszerű mátrixműveletek (transzponálás is megengedett) segítségével a következő mátrixokat:

W =

2	0	0	1	1
0	2	0	1	1
0	0	2	1	1
1	1	1	0	0
1	2	3	0	0
1	3	6	0	0

Z =

1.0000	0.5000	0.3333	0.2500	1.0000
0.5000	0.3333	0.2500	0.2000	1.0000
0.3333	0.2500	0.2000	0.1667	1.0000
0.2500	0.2000	0.1667	0.1429	1.0000
0	0.5000	0	0	0
0	0	0.5000	0	0
0	0	0	0.5000	0
0	0	0	0	0.5000

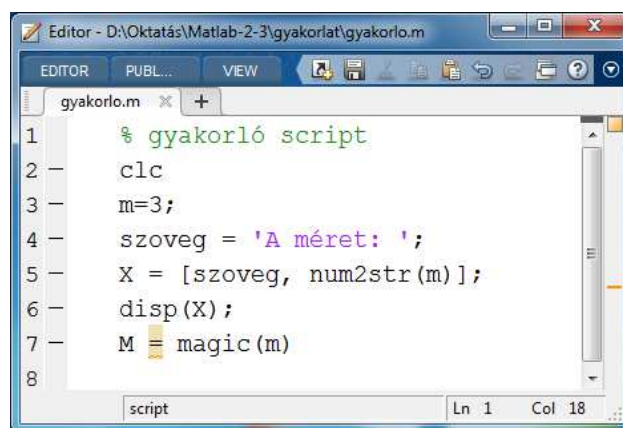
Mágikus mátrix

magic(n) – mágikus négyzet, ahol n legalább 3 és az elemek mind különbözők 1-től n^2 -ig. Az elemek összege a sorokban, oszlopokban, a főátlóban és az antiátlóban megegyezik.

Feladat: Mennyi a 4×4-es mágikus mátrix egy oszlopának elemösszege?

Feladat: Készítsünk scriptet, amely kiírja az $n \times n$ -es mágikus mátrixot és n értékét a képernyőre! Használjuk a *disp* parancsot. (A méret fixen megadható a scriptben, mert bekérést még nem tanultunk.)

Egy lehetséges megoldás:



```
1 % gyakorlo script
2 clc
3 m=3;
4 szoveg = 'A méret: ';
5 X = [szoveg, num2str(m)];
6 disp(X);
7 M = magic(m)
8
```

MÁTRIXFÜGGVÉNYEK

A mátrixmanipulációs függvények (tükrözés, forgatás, eltolás) látványos lehetőségeket adnak az eddig megismerteken túl további származtatott mátrixok előállítására, ill. újabb, bővített „építkezésre”.

Az általunk felhasznált „repertoár”: *flipud*, *fliplr*, *rot90*, *circshift*.

Feladat

Hozzunk létre olyan 5×5-ös méretű mátrixot, amely az antiátlójában csupa 5-ös elemet tartalmaz, a többi pozícióban pedig 0 áll.

Feladat

Készítsük el az 5×5 -ös mágikus mátrixot!

Forgassuk el 90 fokkal az óramutató járásával ellentétesen, majd transzponáljuk!

Adjuk meg a 3. sor 2. elemét!

Feladat

Igazoljuk kétféle módon is, hogy a mágikus mátrix sor- és oszlopösszege megegyezik.

Kidolgozott mintafeladat

Rakjuk össze a speciális mátrixfüggvények segítségével előállított almátrixokból a következő mátrixot!

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

Megoldás

A mátrixunk 4 blokkra bontható, ezek mérete rendre 3×3 , 3×4 ; 4×3 , 4×4 .

A nem négyzetes blokkok zérusmátrixok, a négyzetes mátrixoknak pedig csak az antiátlója tartalmaz egyeseket. Ez utóbbiakat az egységmátrix tükrözésével, vagy forgatásával állíthatjuk elő. A klasszikus összerakós megoldás:

```
>> A = [flipud(eye(3)), zeros(3,4); zeros(4,3), rot90(eye(4))]
```

Egy másik megoldás:

```
>> AE4 = rot90(eye(4)); A(1:3,1:3) = AE4(2:4,1:3); A(4:7,4:7) = AE4
```

Itt felhasználtuk azt, hogy a célmátrix fel nem töltött elemei automatikusan nullák lesznek.

Még rövidebben (mátrixmanipulációs függvényekkel):

```
>> A = flipud(circshift(eye(7),4))
% A 7x7 egységmátrix sorait 4-el ciklikusan előre toltuk, majd
% tükröztük, vagy:
>> A = circshift(flipud(eye(7)),3)
% először tükrözzük, azután shiftelünk
```

Feladat

Oldjuk meg a kidolgozott mintafeladatot úgy is, hogy nem az antiátlókban, hanem a főátlókban vannak az egyes elemek.

OTTHONI MUNKA

Feladat (mátrixszorzás)

Az A 2×2 méretű mátrixszal műveleteket végeztünk, és a következőket kaptuk:

```
>> B = diag(A+A')
```

B =

10

8

```
>> C = A^2
```

C =

41 18

72 32

Mi volt az A mátrix?

(Segítség: az $A + A'$ mátrix szimmetrikus és így főátlójában az A diagonálisának dupla értékei szerepelnek. Az A mátrix hiányzó értékeit *jelöljük p-vel és q-val, majd ezek felhasználásával írjuk fel a C mátrixot*. A konkrét értékekkel való összehasonlításból (papír-ceruza módszer) megkapjuk a keresett adatokat.)

Feladat (speciális szorzatmátrixok)

Legyen L egy valós elemű alsó háromszögmátrix. Ellenőrizzük, hogy az $L * L'$ szimmetrikus-e? („Táltosoknak”: igaz-e ez mindig?)

Feladat (nevezetes mátrixok)

Legyen $A = 1$. Hajtsuk többször végre a következő értékadásokat (mátrixbővítést).

Magyarázzuk meg az eredményt! Egy indexhivatkozásban az *end* szó az index maximális értékét jelenti, lásd `size(A, 1)`, illetve `size(A, 2)`.

```
>> A(end+1, :)=0, A(:, end+1)=0, A(end, end)=1
```

Feladatok (nevezetes mátrixok)

a./ Tördeljük fel 3×3-as mátrixra (*reshape* művelet) a következő sorvektort! Milyen mátrixot kaptunk?

```
>> A = [ones(1, 3) zeros(1, 3) ones(1, 3) zeros(1, 3) ones(1, 3)]
```

b./ Legyen $A = 1$. Hajtsuk többször végre a következő értékadást (mátrixbővítést).

Magyarázzuk meg az eredményt!

```
>> A = [A zeros(size(A, 1), 1); zeros(1, size(A, 2)) 1]
```

Tipp: rajzoljuk le a keletkező új mátrixokat!

Feladat (nevezetes mátrixok, véletlenszámok)

Ismerkedjünk meg a *randn* függvénnyel!

`randn(méret)` – standard normális eloszlású véletlenszámok

```
>> x = 2*randn(400,1) + 10; hist(x)
% 400 darab 10 várható értékű és 2 szórású szám, hisztogram
>> x = rand(400,1); hist(x)
% egyenletes eloszlás illusztrációja hisztogramon
```

Feladat (véletlenszámok)

Ellenőrizzük, hogy sok (itt 192 darab) egyenletes eloszlású véletlen szám összege már más eloszlású lesz!

Tipp: a `sum(A)` művelet az A mátrix oszlopainak összegét képezi egyetlen sorvektorba. Erre kérjünk hisztogramot.

Megoldás: `A = rand(192, 400); x = sum(A); hist(x)`

Feladat (vektorból származtatott mátrixok)

`vander(x)` – az x vektorból előállított Vandermonde mátrix oszlopaiban az x oszlopvektorra alakítottjának a v vektornak pontozott egész hatványai vannak:

$v(i, j) = x(i)^{(n-j)}$, ahol $n = \text{length}(v)$.

Egy Vandermonde mátrix determinánsa az x vektor elemeiből előállított csökkenő indexű összes különbségpár szorzata. Tehát ha minden x elem különböző, akkor a determináns nem lehet 0.

Példa

```
>> x = 1:4, v = vander(x), d = det(V)
```

Feladat

Számítsuk ki a fenti V mátrix determinánsát az adott $(4-3) \cdot (4-2) \cdot \dots \cdot (2-1)$ módon!

Feladat (oszlophivatkozás, oszlopösszeg; nevezetes mátrixokkal)

Lineáris algebrából tudjuk, hogy egy mátrix valamely oszlopának (ill. sorának) összege egységvektorok (e_i) és csupa egyes vektorok segítségével (megfelelő transzponálás műveletek felhasználásával) is előállítható („kiszedhető”). Példaként adjuk meg a következő táblázat Győrtől vett távolsági adatait tartalmazó oszlopát, és a távolságok összegét; majd ismételjük meg ugyanezt a Budapestről mért távolságokra!

	A	B	C	D	E	F	G	H
1		Pécs	Győr	Miskolc	Gyula	Dabas		Kínálát
2	Tatabánya	180	60	260	300	90		50
3	Budapest	160	120	200	220	40		40
4	Udvarhely	600	780	560	450	600		30
5	Szerdahely	240	60	350	400	190		20
6								
7	Kereslet	30	28	5	17	20		
8								

Útmutató (Győr): $A \cdot e_2$ kiszedi a Győrtől mért távolságokat, $o \cdot A \cdot e_2$ pedig előállítja a megfelelő összeget (itt e_2 egységvektor, o pedig csupa egyes vektor, megfelelő méretekkel).

Feladatok (nevezetes mátrixok)

a./ Előkészítésként nézzük meg a `chol` parancs leírását!

Határozzuk meg a 6×6-os Pascal-mátrix Cholesky-felbontását (felső háromszög mátrix, U). Ellenőrizzük, hogy U^*U valóban kiadja a Pascal-mátrixot!

b./ Határozzuk meg a `pascal(5)` és a `pascal(6)` mátrixok determinánsát! Milyen következtetés adódik ebből az inverz mátrixok elemeire? (Ellenőrizzük!)

Feladat (mátrixépítés)

Készítsünk egyedi tervezésű mátrixokat a kidolgozott mintafeladathoz hasonlóan a Matlab speciális mátrixfüggvényeinek felhasználásával (építkezés).

Próbáljunk ki olyan építési feladatokat is, hogy az összerakott mátrix valamely részmátrixát felülírjuk egy új mátrixdarabbal. Pl.

W =

2	0	0	1	1
0	2	0	1	1
0	0	2	1	1
1	1	1	0	0
1	2	3	0	0
1	3	6	0	0

Felülírással módosítva:

W =

2.0000	0	0	1.0000	1.0000
0	1.0000	0.5000	0.3333	1.0000
0	0.5000	0.3333	0.2500	1.0000
1.0000	0.3333	0.2500	0.2000	0
1.0000	2.0000	3.0000	0	0
1.0000	3.0000	6.0000	0	0

Feladat (mátrixfüggvények)

Állítsuk elő a következő (még nem létező) mátrixot elemeire történő értékadásokkal! (lehetőleg a legkevesebbel)

0	10	0	0	0	0	0	0	0	0
0	0	10	0	0	0	0	0	0	0
0	0	0	10	0	0	0	0	0	0
0	0	0	0	0	0	0	0	10	0
0	0	0	0	0	0	0	0	0	10
0	0	0	0	0	0	0	0	0	0

Segítség:

a./ `clear A; A(1:9, 2:10) = 10*eye(9)` az utolsó sor nélküli eredményt adja. Hogyan jöhet létre az utolsó sor?

b./ Shifteljük az egységmátrix 10-szeresét függőlegesen negatív irányba, vagy vízszintesen pozitív irányba, és a felesleges elemet nullázzuk ki!

