



8–9. előadás

Matlab 2–3.

(Mátrixok, hivatkozások, műveletek)

Dr. Szörényi Miklós,
Dr. Kallós Gábor

2014–2015





Tartalom

- Példa: mágikus mátrix
- Mátrixok létrehozása
 - Elemenkénti megadás, Variable editor használata
 - Összeállítás már létező mátrixokból
 - Beolvasás fájlból
- Mátrixok elemhivatkozásai
 - Egyetlen elem, több elem, összefüggő tartomány
- Műveletek mátrixokkal
 - Skalárműveletek
 - Összeadás, kivonás, szorzás, hatványozás
 - Osztás, mátrix inverze
 - Pszeudoinverz, bal- és jobbinverz
 - Logikai műveletek mátrixokkal
- Speciális/nevezetes mátrixok
- Mátrixmanipulációs függvények
- Parancsok archiválása, m-fájlok

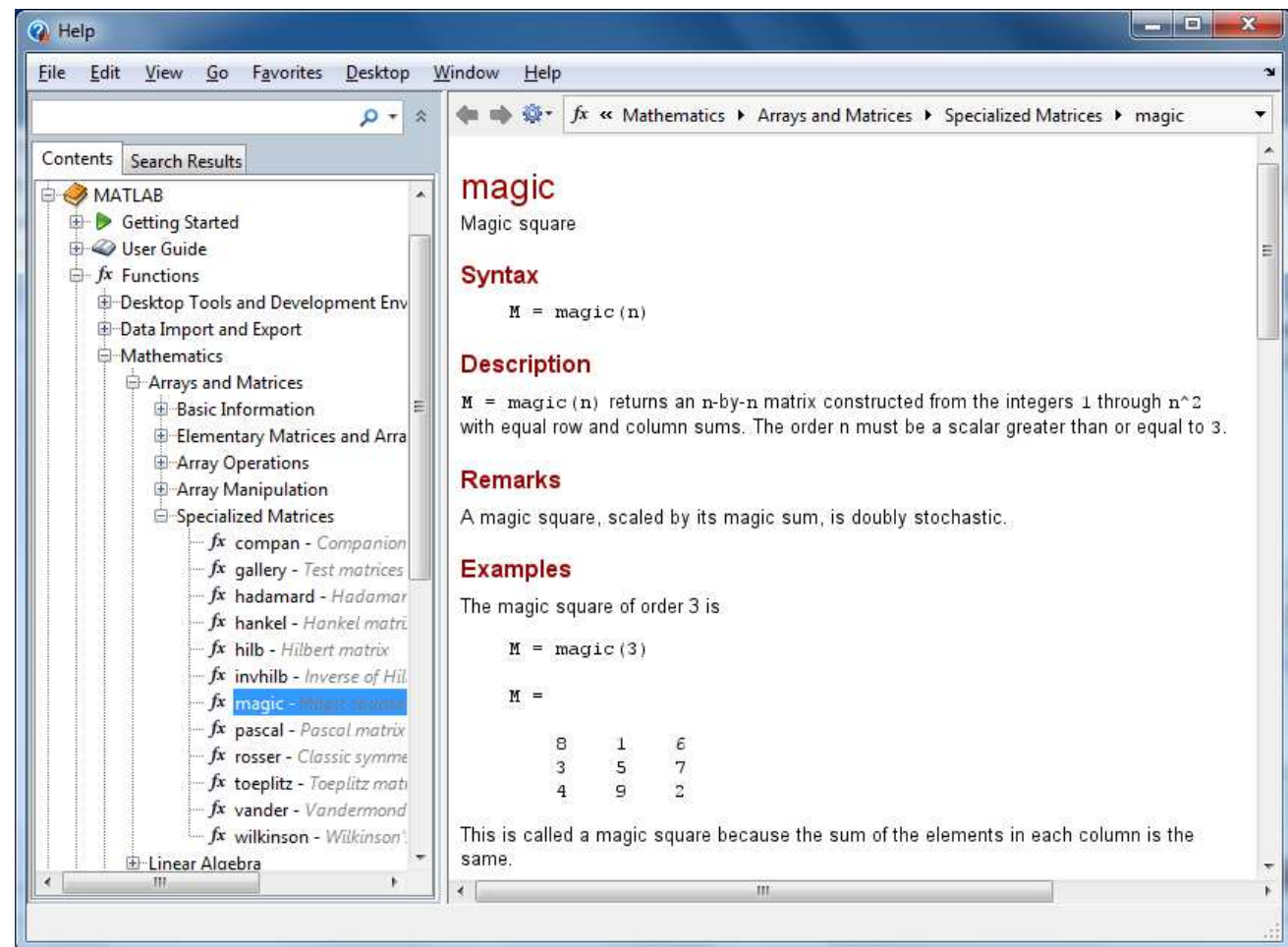


Mátrixok – bevezető

- Egy speciális $n \times n$ -es mátrix: a mágikus mátrix (beépített függvénnyel)
 - Poz. egész számokból épül fel (1-től n^2 -ig)
 - Elemeinek összege egy soron és egy oszlopon belül ugyanaz, és az átlókban is

- Példa

```
>> A = magic(4)
>> sum(A,1)
% oszlopok
% összege
>> sum(A,2)
% sorok összege
>> sum(diag(A))
% főátló
>> sum(diag ...
(flipud(A)))
% antiátló
>> sum(1:16)
% magyarázat:
mágikus összeg
```



Mátrixok – bevezető*

- A 4×4-es mágikus mátrix Dürer rézkarcán (Melankólia)

```
>> load durer % lehet: whos
>> image(X)
% előtte lehet: load detail
>> colormap(map)
>> axis image
```

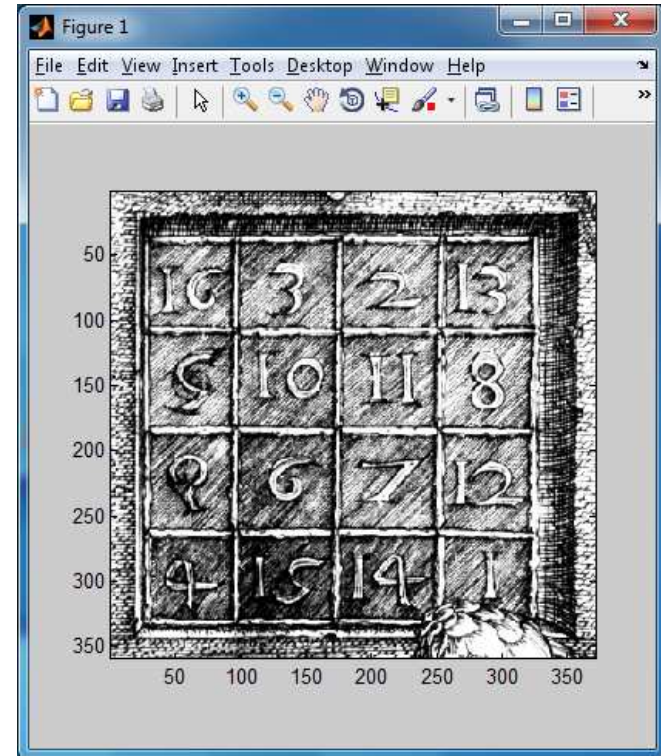
- Dürer mátrixa a Matlab mátrixából a középső két oszlop felcserélésével áll elő

```
>> A = A(:, [1 3 2 4])
% ez általában megváltoztatja az
átlók összegét!
```

- Feladat: Hány db különböző 3×3-as mágikus mátrix létezik? (Forgatások, tükrözések)
Állítsuk elő őket!

```
>> B = magic(3)
>> for k = 0:3
    rot90(B, k)
    rot90(B',k)
end
```

- *Feladat: Állítsuk elő az összes 4×4-es mágikus mátrixot (880 db)





Mátrixok létrehozása

Elemenkénti megadás

- Felsoroljuk az összes elemet
 - $n \times m$ -es mátrixok mellett sorvektorok és oszlopvektorok is ugyanígy megadhatók
 - Üres mátrix is képezhető
 - Az elemek szabállyal is leírhatók, így pl. a `:` operátor is használható
 - Eml.: transzponálás művelet (főátlóra tükrözés)

Példák

```
>> A = [1, 2; 3, 4], B = [1 2; 3 4 5]
% a méretmegadásnak konzekvensnek kell lenni
>> b = [1 2 3 4], c = [1 : 0.1 : 1.4], d = [1:0]
% sorvektorok, üres is lehet!
... d = Empty matrix: 1-by-0
>> d = [1; 2; 3; 4; 5], e = [1*i, -2*i, 3+i, 4-i, 5]'
% oszlopvektorok
>> C = [], d = [6:3]'
% üres mátrixok (0 sor, 0 oszlop, ill. 0 sor 1 oszlop)
```

Tudjuk már

- Mátrix típusa az elemeinek a típusa (class fv.)
- Mátrix méretei lekérdezhetők (size fv.)

Kérdés:

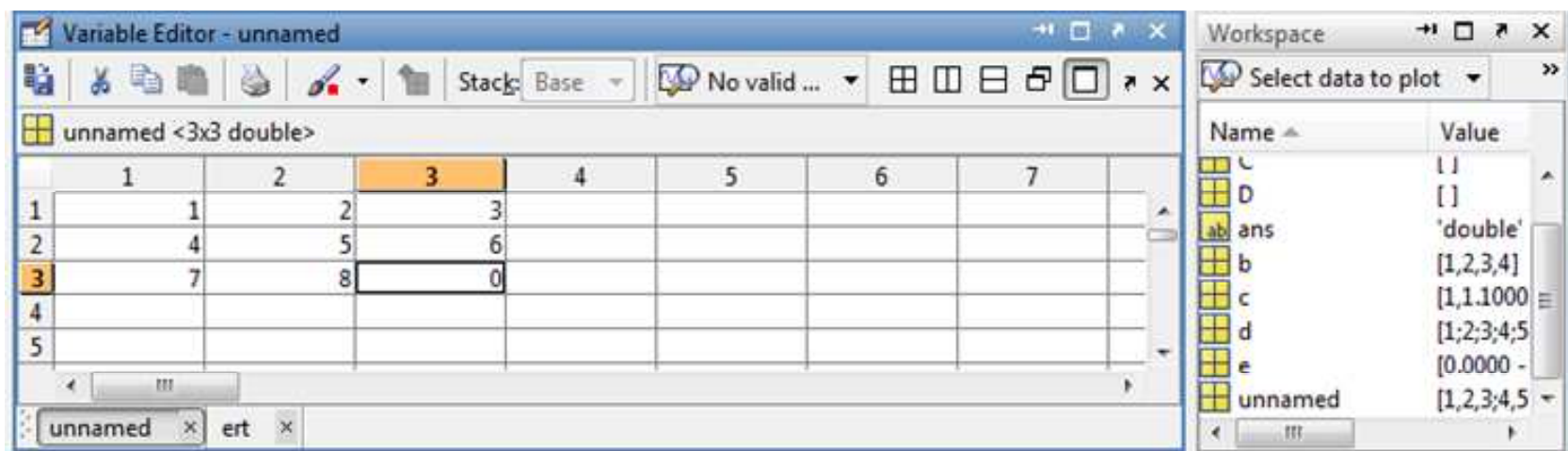
- Mi a fenti A mátrix típusa? Mi az eredmény, ha $A = [1, \text{uint8}(2); 3, 4]$?



Mátrixok létrehozása

Feltöltés a Variable Editor segítségével

- Workspace ablakban Ctrl-N (Variable editor megnyitás)
 - Egy „unnamed” mátrix keletkezik, kezdetben 0 számértékű és 1×1-es
 - Célszerűen: az unnamed mátrix átnevezendő
 - (Már elkészült mátrixok, vektorok is módosíthatók a Variable editorral)
- Adatok begépelése (módosítása)
 - Excel táblázat-szerű környezet
 - Egyszerű, barátságos kezelőfelület, szerény formázási támogatás is elérhető
- Mentés (ha szükséges)
 - .mat fájl keletkezik (kódolt formátum)





Mátrixok létrehozása

Összeállítás már létező mátrixokból

- „Építkezés” összeillesztéssel

- Figyelnünk kell a megfelelő méretekre (!)

- Példa

```
>> A = [1, 2; 3, 4], B = [2.5 ; 4.5]
```

```
>> C = [A B]
```

```
% C mátrixra átöröklődik B megjelenítése
```

- Kérdés:

- Mit kapnánk, ha a $C = [A; B]$ paranccsal illeszténénk össze a mátrixokat?
- Hogyan módosítsuk a B mátrixot, hogy helyes legyen az előző parancs?

- Létrehozás bővítéssel (mint fent)

Példa

```
>> F = [A [10; 20]] % merge: oszlopokkal bővítés
```

```
>> G = [A; [10, 20]] % append: sorokkal bővítés
```

- Még nem létező indexű elemre vonatkozó értékadás is bővíti a mátrixot

```
>> A(3, 3) = 7 % Nézzük meg az új A mátrixot!
```

- Ugyanígy teljesen új mátrix is létrehozható

```
>> H(3:4, 5) = [11; 22]
```

```
% ha egy elem nem kap értéket, automatikusan 0 lesz
```

- Felépítés darabolással

```
>> D = A(1:2,1) % vagy: D = A(:,1), teljes részhev.
```





Mátrixok létrehozása

Beolvasás szöveges fájlból (csv, txt, ...)

- Példa (lineáris egyenletrendszer megoldása)

```
>> load Ab.csv, Ab
% az egyenletrendszer teljes 3x4-es táblázata
>> A = Ab(:, 1:3), b = Ab(:, 4)
% kiemeljük/létrehozzuk az A és b mátrixokat
>> x = A\b, A*x
% megoldjuk az egyenletrendszert, ellenőrzés
```

Mátrix beolvasása Excel munkafüzetből (xlsread)

- Olvassuk el először a *doc xlsread* instrukcióit!
- Az első lapról olvasunk, ill. a munkafüzetnek egyetlen lapja van
 - A bal felső sarokban találhatóak az adatok:
>> A = xlsread('adatok.xls')
- A munkafüzetnek több lapja van, és résztartományból olvasunk
 - A 2. füzetlapról olvasunk:
>> A = xlsread('minta_kész1.xls', 2, 'B3:E6')
 - Nevesített füzetlapról olvasunk:
>> b = xlsread('minta_kész1.xls', '2.feladat', 'G3:G6')





Mátrixok elemhivatkozásai

- Egyetlen elem // $A(s_ind, o_ind)$ ill. $A(sorszám)$
 - Értékadás

```
>> A = ones(4), A(2, 3) = 8
% csak a 2. sor 3. eleme változik, A(10) = 8 is jó lenne
(bármely elemet megváltoztatva a rendszer kiírja a teljes
mátrixot)
>> A(5, 2) = 10
% mint korábban, az 5. sorral bővül, minden nem definiált
elem 0 lesz
```
 - Hivatkozás

```
>> p = 2; q = 3; A(4, 4) = A(p, q) + 1
```
- Több elem, összefüggő tartomány
 - Sorvektor

```
>> a = [4, 7, 2, 1, 8, 3], a(2:4) = 0
% több elemre értékadás egyetlen értékkel -> mind ez lesz
>> a([5, 6]) = [10, 20]
% több elemre több értékkel, az elemszámoknak egyezni kell!
```
 - Sorvektor példa folyt.

```
>> a(1) = []
% első elem fizikai törlése, a vektor mérete eggyel csökken!
```





Mátrixok elemhivatkozásai

■ Több elem, összefüggő tartomány (folyt.)

■ Oszlopvektor

```
>> a = [4; 7; 2; 1; 8; 3], a(2:4) = 0, a(5:6) = [10, 20]  
% oszlopvektor esetén nem kell ' jelet használni (de lehet)!
```

■ Összefüggő tartomány (részmátrix)

Teljes sor // `A(s_ind, :)`

Teljes oszlop // `A(:, o_ind)`

Valódi részmátrix // `A(s_ind_1:s_ind_2, o_ind_1:o_ind_2)`

```
>> P = pascal(6)
```

```
>> P(1:2, [5:6]) = zeros(2)
```

```
% a jobb felső sarok (2x2-es rész) nullázódik
```

```
% itt a P([1:2], 5:6) hivatkozás is jó! (a többi változat is)
```

■ Részek másolása, felülírása

```
>> P(:, 3:7) = P(:, 2:6)
```

```
% 2-6 oszlopok eltolása a 3-7 oszlopokba  
(nem veszünk értéket, a mátrix mérete nő)
```

```
>> P(3:7, :) = P([2:6], :)
```

```
% 2-6 sorok eltolása a 3-7 sorokba
```

■ Részek törlése – mint korábban

(Vigyázzunk arra, hogy mátrixból fizikailag csak teljes sor vagy oszlop törölhető!)

```
>> P(:, 2) = 0 % 2. oszlop nullázása
```

```
>> P(2, :) = [] % 2. sor fizikai törlése
```





Mátrixok elemhivatkozásai

- Komplex vektorok/mátrixok is ugyanígy kezelhetők

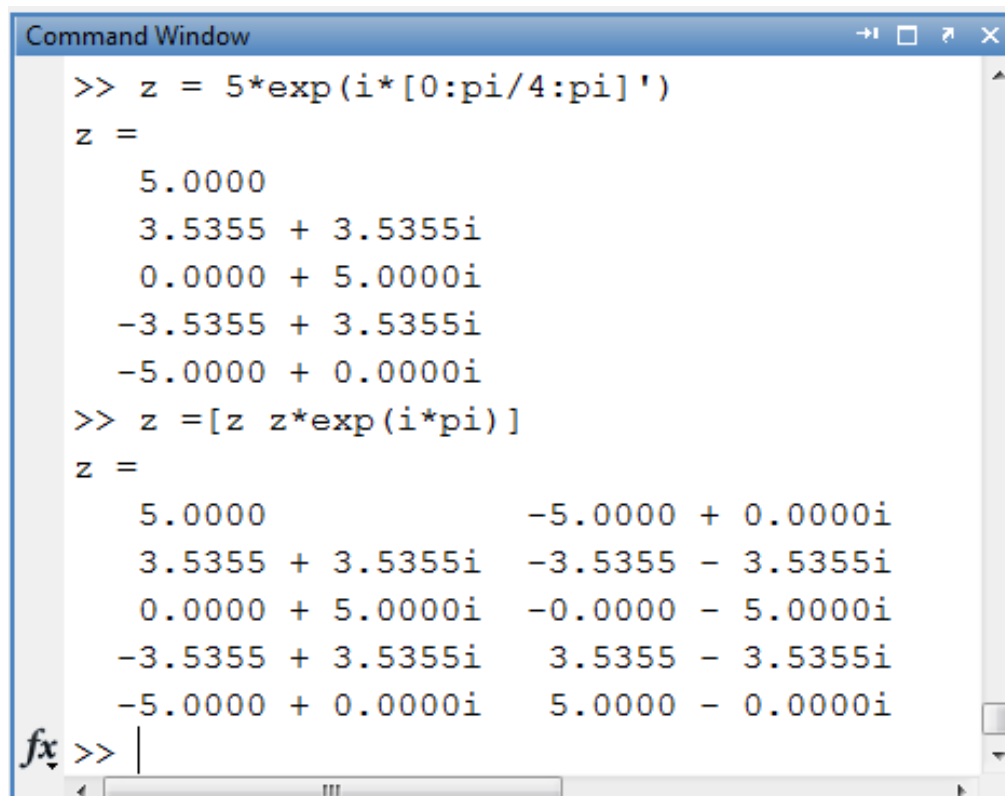
- Példa: komplex sorvektor létrehozása és több lépéses bővítése

```
>> z = 5*exp(i*[0:pi/4:pi]')
% 45 fokenkénti komplex számok (lépésköz)
>> z = [z z*exp(i*pi)]
% merge, 180 fokkal elforgatottakkal bővítés
>> z = [z; z]
% append, saját magával való bővítés
```

- Speciális lehetőség: átméretezés

- Adott sor/oszlopvektor ($n \times m$ elem) a Matlab rendszerben $n \times m$ -es (ill. $m \times n$ -es) mátrixszá alakítható át
- Példa

```
>> v = [1 2 3 4 5 6]
>> A = reshape(v, 3, 2)
>> A = reshape(v, 2, 3)
% Figyeljük a méreteket!
```
- Feladat: próbáljuk ki ezt úgy is, hogy v oszlopvektor!
Mi változik?



```
Command Window
>> z = 5*exp(i*[0:pi/4:pi]')
z =
    5.0000
    3.5355 + 3.5355i
    0.0000 + 5.0000i
   -3.5355 + 3.5355i
   -5.0000 + 0.0000i
>> z = [z z*exp(i*pi)]
z =
    5.0000           -5.0000 + 0.0000i
    3.5355 + 3.5355i   -3.5355 - 3.5355i
    0.0000 + 5.0000i   -0.0000 - 5.0000i
   -3.5355 + 3.5355i    3.5355 - 3.5355i
   -5.0000 + 0.0000i    5.0000 - 0.0000i
fx >> |
```





Műveletek mátrixokkal (operátorok)

Skalárműveletek

- A mátrix minden elemén (elemenként) végrehajthatódnak
 - Kivéve: mátrix hatványozása, ld. lent
- Összeadás, kivonás

```
>> A = [1, 2; 3, 4]
>> A + 2
```

 - $A + s = s + A$ érvényes
- Skalárral való szorzás

```
>> A*0.5
```

 - A művelet pontozott változata is végrehajtható (ugyanaz)
 - $A*s = s*A$ érvényes
- Skalárral való osztás
 - Mint a szorzás
- Hatványozás
 - Négyzetes mátrix esetén – esetleg ismételt – mátrixszorzás hajtható végre (ott nézzük meg; pl. A^3)
 - A művelet pontozott változata is végrehajtható (elemenként hatványoz), nem kell, hogy négyzetes legyen a mátrix

```
>> A.^2
```

```
Command Window
>> x=0:0.2:1
x =
    0    0.2000    0.4000    0.6000    0.8000    1.0000
>> y=x.*x
y =
    0    0.0400    0.1600    0.3600    0.6400    1.0000
>> y=x.^2
y =
    0    0.0400    0.1600    0.3600    0.6400    1.0000
fx >>
```





Műveletek mátrixokkal

Összeadás, kivonás

- Csak azonos méretű mátrixok/vektorok adhatók össze, vagy vonhatók ki!
 - Méreteltérésnél: hibaüzenet
- $A +$ illetve $-$ művelet elempáronként hajtódik végre!
 - Nincs pontozott művelet, azaz $.+$ vagy $.-$ nem írható! (hibát kapunk)
- Példa

```
>> A = ones(2,3), A2 = A + A, A1 = A2 - A
>> B = [1 2 3 4] + [5, 6, 7, 8]
```

Transzponálás

- A transzponálás főátlóra tükröz és komplex esetben konjugál is
 - Azaz tdk.: az oszlopok és a sorok felcserélődnek
 - Tudjuk: oszlopvektor transzponáltja sorvektor és fordítva

```
>> x = [0:0.1:1]'
```
 - Jelölés: A' vagy A^*
- Pontozott formában csak transzponál, konjugálás nincs

```
>> T = exp(i*[0:pi/3:pi]')
>> T1 = T'
>> T2 = T1.'
```





Műveletek mátrixokkal

Szorzás

- Mikor végezhető el a művelet?
 - Rajzoljunk... (gondoljuk végig)
 - Méretszabályok: $[m; n] \times [p; q] \rightarrow [m; q]$ és $n = p$ (első méret: sorok száma)
- (A mátrixok szorzása „nagyon nem kommutatív” művelet)
 - Előfordulhat, hogy $\mathbf{A} \cdot \mathbf{B}$ végrehajtható, de $\mathbf{B} \cdot \mathbf{A}$ nem; lehet az is, hogy $\mathbf{A} \cdot \mathbf{B} \neq \mathbf{B} \cdot \mathbf{A}$
- Matlabban: „sima” * operátor
 - Méreteltérésnél: hibaüzenet
- Példa

```
>> ones(2,3)*ones(3,4)      % összeszorozható, méret 2x4 lesz
>> ones(3,2)*ones(4,3)      % hibaüzenet jön:
??? Error using ==> mtimes
Inner matrix dimensions must agree.
```
- Pontozott szorzás: azonos méretű mátrixok megfelelő elempárjai szorzódnak

```
>> S = 2*ones(2,3), R = [1:3; 4:6], Q = S.*R
```

Hatványozás

- Csak négyzetes mátrixokra, a pontozott változat is használható; példa:

```
>> A = [1 2; 2 1], A_2 = A^2, A_3 = A^3, A3 = A.^3
>> E = eye(3), n = [0.6 0.8 0], P = E - n'*n
% projektormátrix (vetítés)
>> P^2, P^7 % projektormátrix akárhányadik hatványa is P
```





Műveletek mátrixokkal

Osztás, mátrix inverze

- Inverz mátrix (csak négyzetes mátrixokra)
 - Az a mátrix, amelyet az eredetivel összeszorozva egységmátrixot kapunk
 - Jelölés: \mathbf{A}^{-1}
 - Érvényes: $\mathbf{A} \cdot \mathbf{A}^{-1} = \mathbf{A}^{-1} \cdot \mathbf{A} = \mathbf{E}$
 - Előállítás Matlabban: `inv` parancs
 - Egységmátrix: a főátlóban csupa 1-es, más helyeken 0-ák
 - Jelölés: \mathbf{E} vagy \mathbf{I}
 - Az egységmátrixra teljesül: $\mathbf{A} \cdot \mathbf{E} = \mathbf{E} \cdot \mathbf{A} = \mathbf{A}$; $\mathbf{E} \cdot \mathbf{E} = \mathbf{E}$
 - Előállítás Matlabban: `eye` parancs
 - Inverz akkor létezik, ha a *determináns nem 0*, illetve a mátrix rangja megegyezik a sorainak (oszlopainak) számával
- Mátrixszal való osztás valójában az inverzzel történő szorzást jelenti
 - Ilyen példákat már láttunk (Mire jó a Matlab?; v. ö.: Excel feladatok)
 - Eml.: Az egyenletrendszer $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ alakban adott, és ha $\det(\mathbf{A}) \neq 0$, akkor
$$\mathbf{A}^{-1} \cdot \mathbf{A} \cdot \mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b}, \text{ azaz } \mathbf{E} \cdot \mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b}, \mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b}$$
- Ha a determináns 0, vagy a mátrix nem négyzetes, akkor csak a Moore–Penrose-féle pszeudoinverz határozható meg egyértelműen
 - (Ez a mátrix valamilyen értelemben optimális tulajdonságú)



Műveletek mátrixokkal

Osztás, mátrix inverze (folyt.)*

- A Moore–Penrose-féle kváziinverz előállításának bemutatása
 - Ha több (n) sor van mint (m) oszlop ($n > m$), akkor a kváziinverz: $(A' \cdot A)^{-1} \cdot A'$
 - Ennek mérete $m \cdot n$, azaz A' méretével egyezik meg
 - Excelben:
`{=Mszorzat(Inverz.mátrix(Mszorzat(Transzponálás(A); A)); Transzponálás(A))}`
 - Ha kevesebb (n) sor van mint (m) oszlop ($n < m$), akkor a kváziinverz: $A' \cdot (A \cdot A')^{-1}$
 - Ennek mérete is $m \cdot n$, azaz A' méretével egyezik meg
 - Excelben:
`{=Mszorzat(Transzponálás(A); Inverz.mátrix(Mszorzat(A; Transzponálás(A))))}`
- Példa Excelben
 - Most négyzetes a mátrixunk, és a valódi inverz is létezik
- A Matlab is elő tudja állítani a kváziinverz mátrixot (*pinv* parancs, lásd lent)

Microsoft Excel screenshot showing matrix calculations for a 3x3 matrix A.

	A	B	C	D	E	F	G	H	I	J
1			A		b		A ⁻¹			
2										
3		2	2	0	6		-0,25	-0,5	1	
4		1	1	2	9		0,75	0,5	-1	
5		2	1	1	7		-0,25	0,5	0	
6										
7		det(A)		4		ellenőrzés				
8										
9		x = A ⁻¹ b		1		Ax		6		
10				2				9		
11				3				7		
12										
13										
14	A*	2	1	2		AA*	8	4	6	
15		2	1	1			4	6	5	
16		0	2	1			6	5	6	
17										
18	(AA*) ⁻¹	0,6875	0,375	-1		A*(AA*) ⁻¹	-0,25	-0,5	1	
19		0,375	0,75	-1			0,75	0,5	-1	
20		-1	-1	2			-0,25	0,5	0	
21										



Műveletek mátrixokkal

Osztás, mátrix inverze (folyt.)

- A Matlabban van balosztó \ és jobbosztó / művelet is
 - Tehát megoldhatók megfelelő mátrixegyenletek, és meghatározható egy/a bal- és jobbinverz (ha léteznek)
 - Magyarázat →
- A Matlabban ...
 - ... $A \setminus B$ eredménye egy olyan X , amelyre $AX = B$ (ha $B = E$, akkor $A \setminus E$) ...
 - (A és B sorainak száma ekkor kötelezően azonos)
 - ... A / B eredménye egy olyan Y , amelyre $YB = A$ (ha $A = E$, akkor E / B) ...
 - (A és B oszlopainak száma ekkor kötelezően azonos)
 - ... és érvényes $B / A = (A \setminus B)'$ (ha léteznek)
- Példa 1.

```
>> A = [1 2; 3 4], B = [5 4; 3 2], C = A \ B
>> A * C    % ellenőrzés, ez valóban B
>> C * A    % ez viszont már nem B-t adja
```
- Példa 2.

```
>> A = [1 2 3; 4 5 6]    % 3x2-es, rangja 2
>> J = A \ eye(2)        % jobbinverz létezik
>> B = eye(3) / A        % balinverz nem létezik!
```
- Példa 3.
Ha egy mátrixnak egyszerre létezik bal- és jobbinverze, akkor azok megegyeznek, és a mátrix ilyenkor négyzetes (lásd gyakorlat)

Nem négyzetes mátrixok [szerkesztés | forrásszöveg szerkesztése]

A nem négyzetes mátrixok nem invertálhatóak, de létezhet bal- vagy jobbinverzük. Ha az A $n \times m$ -es mátrix rangja n , akkor létezik egy B mátrix, hogy $BA = I$. Ez a B mátrix A balinverze. Hasonlóan, ha az A $n \times m$ -es mátrix rangja m , akkor létezik egy B mátrix, hogy $AB = I$. Ez a B mátrix A jobbinverze.

A Moore-Penrose pszeudoinverz értelmezhető nem négyzetes mátrixokra és nem teljes rangú esetre is. Ennek néhány tulajdonsága megegyezik az inverz tulajdonságaival, és nem szinguláris négyzetes mátrix pszeudoinverze a mátrix inverze.

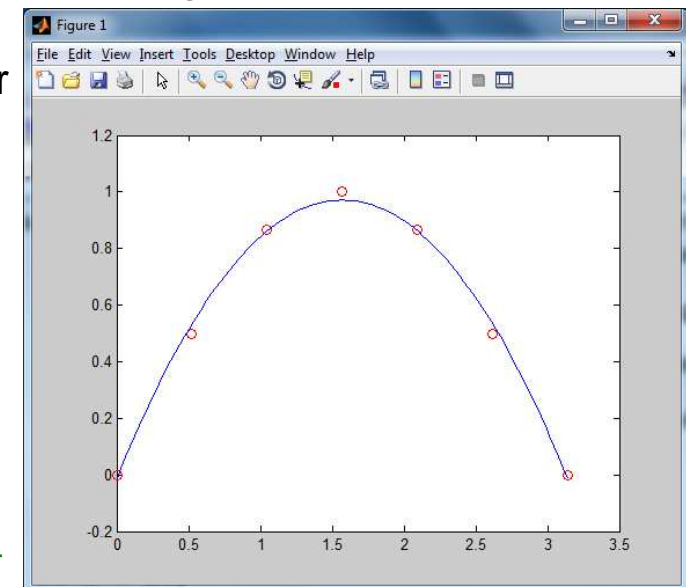


Műveletek mátrixokkal

Osztás, mátrix inverze (folyt.) – alkalmazási példa*

- Gyakori feladat, hogy olyan egyenletrendszert kell megoldani, ahol több az egyenletek száma, mint az ismeretleneké
 - Például a polinomiális regressziónál több síkbeli pont közé legjobban illeszkedő alacsonyabb fokszámú polinomot kell illeszteni
 - Ekkor a megoldandó túlhatározott egyenletrendszer együtthatómátrixa egy Vandermonde mátrix (ld. később) utolsó pár oszlopa
- Pl. illesszünk a sin függvény első félperiódusának 30 fokenkénti pontjaihoz parabolát (másodfokú függvényt)!

```
>> x = 0:pi/6:pi, y = sin(x)
% a 7 darab pont a sin függvénnnyel
>> A = vander(x), A(:, [1:4]) = []
% Vandermonde mátrix utolsó 3 oszlopa
>> c = A\y'
% a másodfokú regressziós polinom együtthatói
>> w = 0:pi/100:pi; f = polyval(c, w);
% polinom pontsorozata (kiértékelés)
>> plot(x,y, 'or', w,f, '-')      % rajzolás
```





Műveletek mátrixokkal

Logikai műveletek mátrixokra

- Függvénnnyel mindig, de általában operátorral is megvalósíthatók (1-2 kivétellel)
- Az $A = [5 \ -2; \ 4 \ 3]$ és $B = [2 \ 4; \ 4 \ -2]$ mátrixokra megnézzük a műveletek eredményét →
- Any és all példa
 $\gg A = [5 \ 2 \ 1; \ 2 \ 6 \ 0; \ 1 \ 0 \ 4],$
 $al = all(A),$
 $an = any(A)$

	$all(A)$	hasonlóan működik, mint az $any()$ értékelés művelete: mind nem 0?	1 1
--	----------	---	-----

Logikai operátor	Függvénnnyel	Jelentés	Eredmény
$==$	$eq(A,B)$	egyenlő	0 0 1 0
$<$	$lt(A,B)$	kisebb	0 1 0 0
$>$	$gt(A,B)$	nagyobb	1 0 0 1
$<=$	$le(A,B)$	kisebb, vagy egyenlő	0 1 1 0
$>=$	$ge(A,B)$	nagyobb, vagy egyenlő	1 0 1 1
\sim	$ne(A,B)$	nem egyenlő	1 1 0 1
$\&$	$and(A,B)$	elemenkénti logikai ÉS	1 1 1 1
$ $	$or(A,B)$	elemenkénti logikai VAGY	1 1 1 1
\sim	$not(A)$	elemenkénti logikai tagadás	0 0 0 0
	$xor(A,B)$	logikai kizáró VAGY	0 0 0 0
	$any(A)$	vektorra: van-e benne nem 0? mátrixra: oszlopokon értékeli ugyanest, sorvektort ad	1 1





Műveletek mátrixokkal

Logikai műveletek mátrixokra (folyt.)

- Speciális eset: logikai műveletek skalárookra (részben már tanultuk)
- True és false előre definiált (védett) változók; értékük 1 ill. 0
 - De: felüldefiniálhatók! (Törlés: clear; tudjuk)
- Érdekesebb példák

```
>> ~2014, ~0
```

```
% nem 0 érték tagadása: 0, a 0 tagadása: 1
```

```
>> 2014 && 1526, 2014 && false, 0 || 2014 % Szerepük:  
vezérlő szerkezetekben (if, while, ...)
```

Logikai operátor	Jelentés
~	numerikus érték logikai tagadása
&&	logikai ÉS (skalárookra)
	logikai VAGY (skalárookra)

```
Command Window
>> 2014 && 1526, 2014 && false, 0 || 2014
ans =
     1
ans =
     0
ans =
     1
fx >> |
```





Speciális mátrixok

- A Matlab rendszerben több speciális mátrix (vektor) *beépített paranccsal* is előállítható
- Alapmátrixok
 - $n \times n$ -es ($n \times m$ -es) egységmátrix: `eye(n)` ill. `eye(n, m)` vagy `eye(méret, típus)`
 - $n \times n$ -es ($n \times m$ -es) csupa egyes mátrix: `ones(n)` ill. `ones(n, m)` [típussal is]
 - $n \times n$ -es ($n \times m$ -es) csupa nulla mátrix: `zeros(n)` ill. `zeros(n, m)` [típussal is]
 - Csupa igaz és hamis mátrix: `true(n)` ill. `true(n, m)` [false is]
- Sorozatok
 - Lineáris sorozat, sorvektor: `linspace(mettől, meddig, mennyi)`
 - Transzponálással oszlopvektor
 - 10-es alapú logaritmikus léptékezésű sorozat (dekádok): `logspace()`
- Nevezetes mátrixok
 - Pascal-háromszög (négyzetes elrendezés, Pascal-mátrix): `pascal(n)` [kód is megadható]
 - Hilbert-mátrix: `hilb(n)`
 - Vandermonde-mátrix: `vander(vektor)`
 - Mágikus mátrix: `magic(n)`
 - Gallery mátrixai: `gallery('mátrixnév', paraméterek)`
- Egyéb csoportok
 - Véletlenszámok: `rand(méretek)`, `randi(határok, méretek)`, `randn(méretek)`
 - Diagonálmátrixok: `diag(v)`, `diag(v,k)`, `diag(X)`, `diag(X,k)`





Speciális mátrixok

Példák

■ Alapmátrixok

```
>> eye(2), eye(2, 3), eye(size(A), 'int16')  
% itt A mérete: 3x3  
>> zeros(2), zeros(2, 3), zeros(size(A))  
>> ones(2), ones(2, 3), ones(size(A))  
>> true(3, 2), false(2, 3)
```

■ Sorozatok

```
>> fi = linspace(0 , 2*pi , 181); kor = exp(i*fi);  
plot(kor), axis square  
>> x = logspace(1, 2, 51), y = log10(x), w =  
exp(log(10)*y)  
% mérnököknek frekvenciaátmenethez
```

```
Command Window  
>> A = ones(3,8)  
A =  
     1     1     1     1     1     1     1     1  
     1     1     1     1     1     1     1     1  
     1     1     1     1     1     1     1     1  
>> B = false(2,5)  
B =  
     0     0     0     0     0  
     0     0     0     0     0  
fx >> |
```



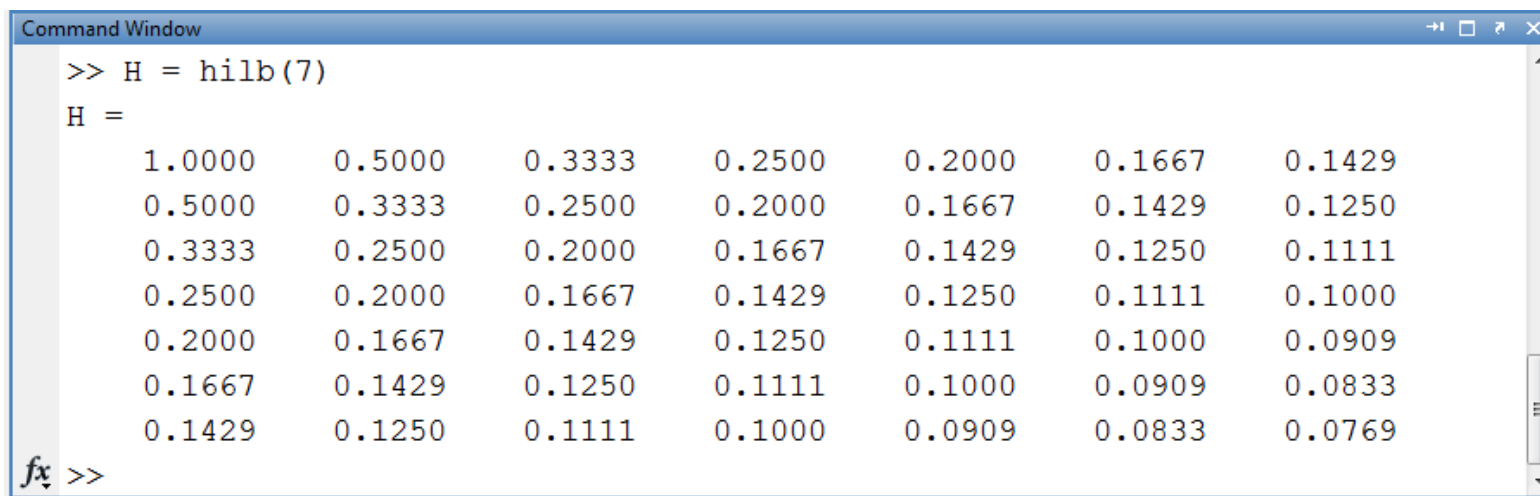


Speciális mátrixok

Példák (folyt.)

- Nevezetes mátrixok

```
>> P6 = pascal(6)
% balról jobbra felfelé menő átlókban a bin. eh-ók
>> P6L = sym(pascal(6,1))
% a Cholesky-felbontás alsó háromszög mátrixa*
>> P6U = P6L', P6L*P6U % előáll a Pascal-mátrix
>> inv_P6 = inv(P6U)*inv(P6L), inv_P6*P6
>> det(P6), det(inv_P6)
% az eredeti és az inverz determinánsa egyaránt 1
>> vander(1:0.5:3)
% oszlopokban a vektor pontozott hatványai
>> H = hilb(5), rats(H), invhilb(5), invhilb(5)*H
% H(i,j) = 1/(i+j-1)
```



```
Command Window
>> H = hilb(7)
H =
    1.0000    0.5000    0.3333    0.2500    0.2000    0.1667    0.1429
    0.5000    0.3333    0.2500    0.2000    0.1667    0.1429    0.1250
    0.3333    0.2500    0.2000    0.1667    0.1429    0.1250    0.1111
    0.2500    0.2000    0.1667    0.1429    0.1250    0.1111    0.1000
    0.2000    0.1667    0.1429    0.1250    0.1111    0.1000    0.0909
    0.1667    0.1429    0.1250    0.1111    0.1000    0.0909    0.0833
    0.1429    0.1250    0.1111    0.1000    0.0909    0.0833    0.0769
```



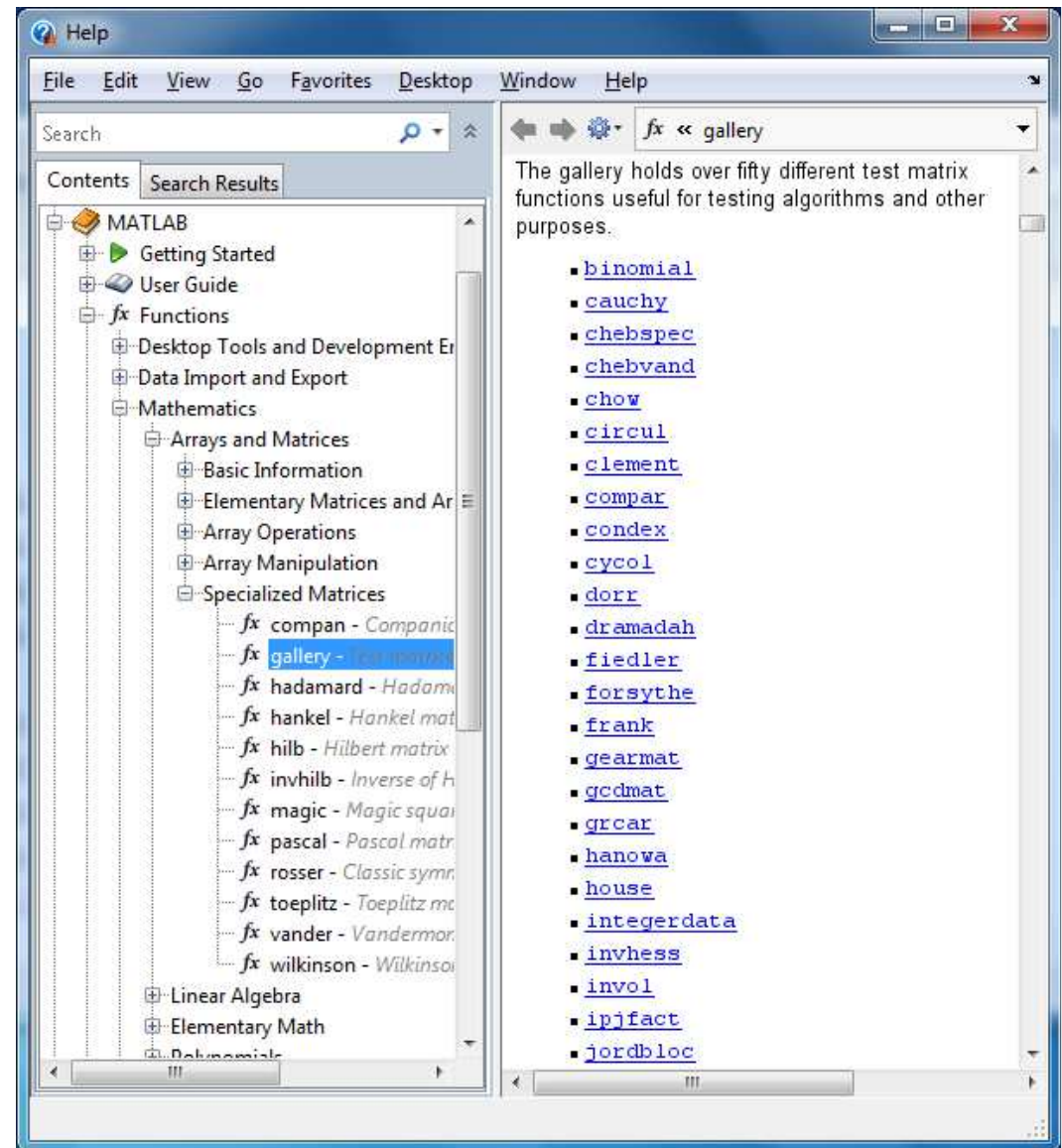
Speciális mátrixok

Példák (folyt.)

- Nevezetes mátrixok/2.

```
>> M4 = magic(4),
sum(M4), sum(M4'),
sum(diag(M4)),
sum(diag(flipud(M4)))
% sorösszegek,
% oszlopösszegek,
% főátló-összeg,
% antiátló-összeg
% egyeznek
```

```
>> n = 4, A =
gallery('binomial',n),
A*A/(2^(n-1)),
B = A*2^((1-n)/2), B*B
% B egy involutórius
% mátrix (inverze saját
% maga)
```

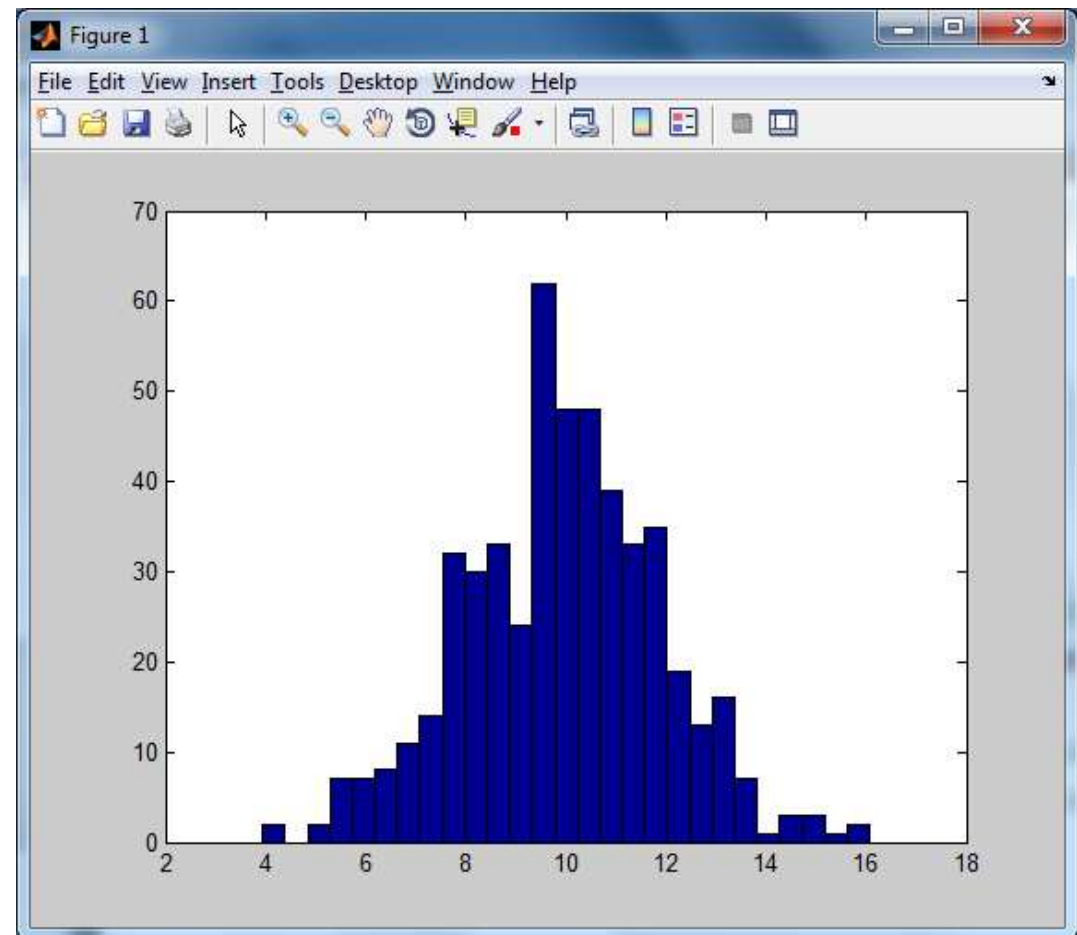


Speciális mátrixok

Példák (folyt.)

- Egyéb csoportok (véletlenszámok)

```
>> uniform = rand(1,6)
% [0..1)-beli hatelemű sorvektor
>> lotto =
randi([1,90], 1, 5)
% lottószámok (de:
% azonosak is lehetnek
% köztük)
>> norm = 10.0 +
2.0*randn(1,500),
mean(norm), std(norm),
hist(norm, 27)
% 500 elemű normális
% eloszlású sorvektor,
% 10.0 várható értékkel
% és 2.0 szórással
% hisztogram 27
% részintervallummal
```





Speciális mátrixok

Példák (folyt.)

- Egyéb csoportok (folyt.)

Diagonálmátrixok: `diag(v)`, `diag(v,k)`, `diag(X)`, `diag(X,k)`

- ```
>> X = diag(lotto), X_2 = diag(lotto,-2),
Y = rand(4), v = diag(Y)', v1 = diag(Y, 1)'
% v - sorvektor, X - mátrix, k - mellékátló sorszám
(negatív is lehet!)
```

```
Command Window
>> X = diag(lotto), X_2 = diag(lotto,-2), Y = rand(4); v = diag(Y)', v1 = diag(Y, 1)'
X =
 26 0 0 0 0
 0 50 0 0 0
 0 0 87 0 0
 0 0 0 87 0
 0 0 0 0 15
X_2 =
 0 0 0 0 0 0 0
 0 0 0 0 0 0 0
 26 0 0 0 0 0 0
 0 50 0 0 0 0 0
 0 0 87 0 0 0 0
 0 0 0 87 0 0 0
 0 0 0 0 15 0 0
v =
 0.3947 0.3309 0.4299 0.8085
v1 =
 0.0196 0.8217 0.3968
fx >>
```





## Mátrixmanipulációs függvények

- Mátrix tükrözése (függőlegesen, vízszintesen):  
flipdim(mátrix, dimenzió), flipud(A), fliplr(A)  
>> A = [1:4; 5:8; 9:12; 13:16], flipdim(A, 1),  
flipdim(A, 2), flipud(A), fliplr(A)  
% A parancsok párosával ekvivalensek
- Mátrix forgatása 90 fokonként balra: rot90(mátrix, hányszor)  
>> rot90(A), rot90(A, 2), eredeti = rot90(A, 4)
- Mátrix körkörös shiftelése: circshift(mátrix, mennyivel)  
>> A = [11:15; 21:25; 31:35; 41:45; 51:55]  
>> circshift(A, 1), circshift(A, -2)  
% függőleges shiftelés  
>> circshift(A, [0 1]), circshift(A, [0 -2])  
% vízszintes shiftelés
- Vektor rendezése: sort(v), sort(v, 'descend')  
% itt v sor, vagy oszlopvektor  
>> v = randi([10 99], [1 6]),  
[rendezett, index] = sort(v, 'descend'), v(index)  
% rendezett == v(index) !!
- Mátrix oszlopainak rendezése: sort(X), sort(X, 'descend')  
>> X = rand(6), sort(X)





## Parancsok archiválása, m-fájlok

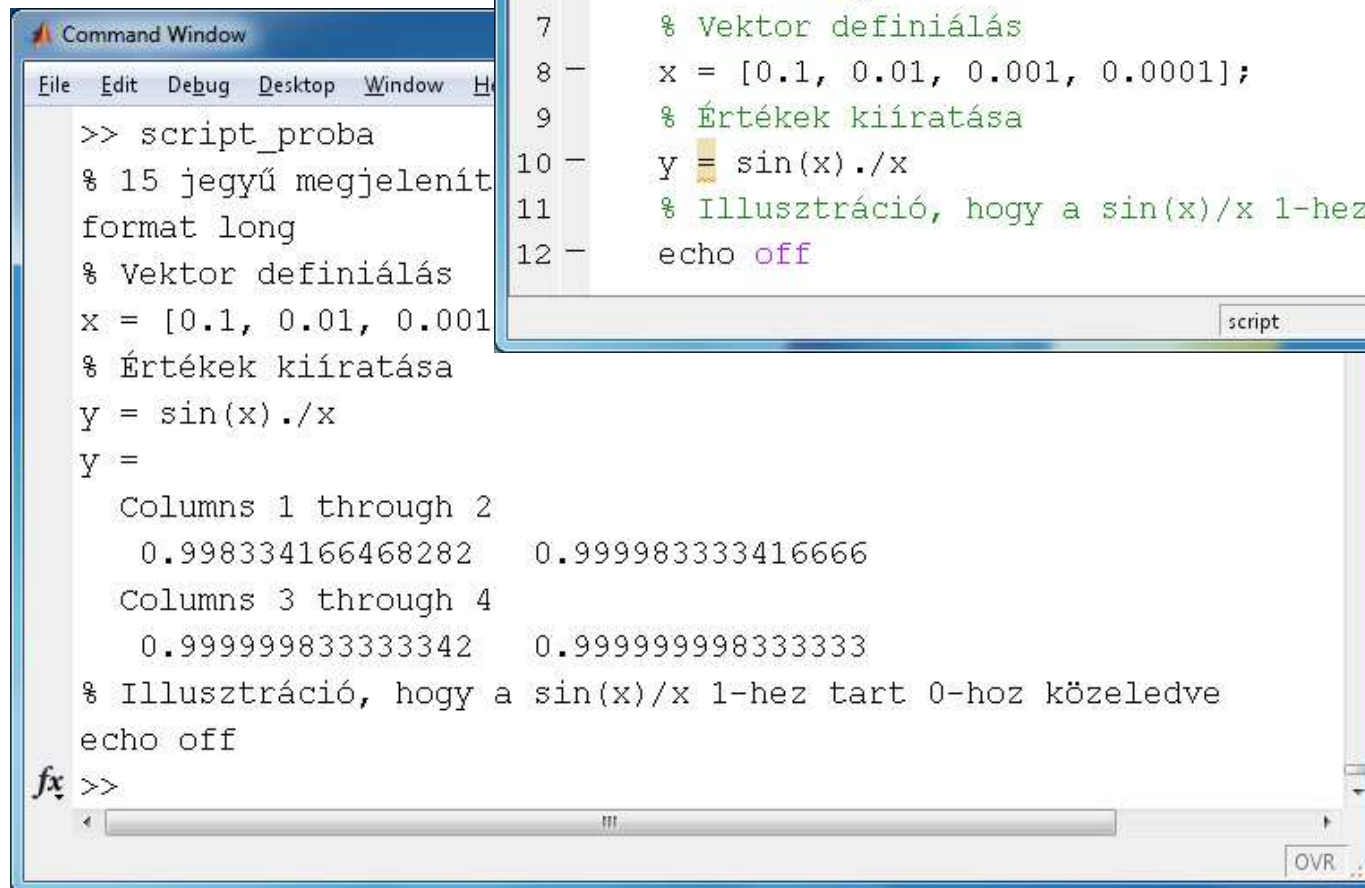
- A Matlabnak szóló és végrehajtott parancsainkat szöveges fájlba menthetjük
- Naplófájl használata
  - diary on – bekapcsolja a parancsablakbeli kommunikáció rögzítését
  - diary off – lezárja a kommunikáció rögzítését
  - Az *elkészült fájl szövegszerkesztővel szerkeszthető*, majd menthető
  - Ha olyan változatot készítünk, amiből töröljük a rendszer válaszait (és a felesleges parancsokat), akkor mentéskor használjuk a .m kiterjesztést!
- Command History
  - Az ablakban a rögzített parancsokból *kijelölhetjük a mentendőket* (Ctrl + egér klikk), majd a helyzetérzékeny menüben a *Create M-file választással menthetjük is* (pl. parancsok.m névvel)
    - Ezzel már egy script keletkezik
- Az m-fájlokról
  - Az aktuális könyvtárban lévő *m-fájl a vezetéknévének begépelésével lefuttatható*
  - Ha saját függvényt írunk, akkor annak a kiterjesztése is .m lesz
    - Ennek a hívása (lefuttatása) a paraméterek helyes megadását, és az eredmények megnevezését kívánja meg (ld. később)





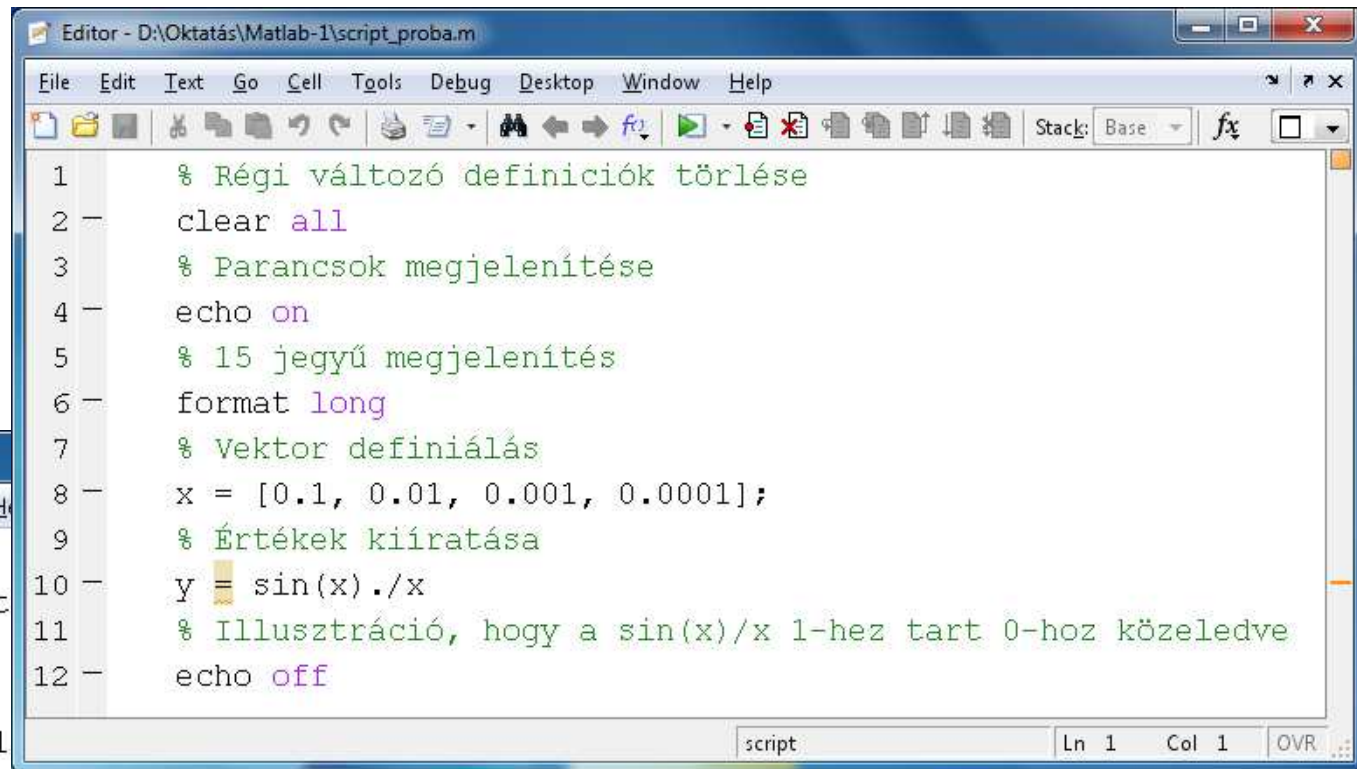
## Script példa

- Script példa –  
tipikusan  
ismétlődő  
tevékenységek



The screenshot shows the MATLAB Command Window with the following output:

```
>> script_proba
% 15 jegyű megjelenítés
format long
% Vektor definiálás
x = [0.1, 0.01, 0.001, 0.0001];
% Értékek kiírása
y = sin(x)./x
y =
 Columns 1 through 2
 0.998334166468282 0.999983333416666
 Columns 3 through 4
 0.999999833333342 0.999999998333333
% Illusztráció, hogy a sin(x)/x 1-hez tart 0-hoz közeledve
echo off
fx >>
```



The screenshot shows the MATLAB Editor window with the following script content:

```
1 % Régi változó definíciók törlése
2 clear all
3 % Parancsok megjelenítése
4 echo on
5 % 15 jegyű megjelenítés
6 format long
7 % Vektor definiálás
8 x = [0.1, 0.01, 0.001, 0.0001];
9 % Értékek kiírása
10 y = sin(x)./x
11 % Illusztráció, hogy a sin(x)/x 1-hez tart 0-hoz közeledve
12 echo off
```