Közelítő szimbolikus számítások

MATLAB

SZTE-TTIK, Számítógépes Optimalizálás Tanszék

Definíció

- Mátrix: a táblázat matematikai fogalma
- "Ha n sorba és m oszlopba rendezünk matematikai jeleket, akkor egy n x m -es mátrixot kapunk."
- Pl.: $A = \begin{pmatrix} 2 & -1 & 0 \\ 3 & 5 & 1 \end{pmatrix}$ 2 x 3 -as mátrix
- Matlabban: `A = [2 -1 0; 3, 5, 1]`
- Mátrix elemek: $a_{1,3} = 0$, $a_{2,2} = 5$ `A(1,3) A(2,2)`
- Az első koordináta a sor a második az oszlop!

Mátrix műveletek

$$A = \begin{pmatrix} 2 & -1 & 0 \\ 3 & 5 & 1 \end{pmatrix} \qquad B = \begin{pmatrix} -1 & 1 & 0 \\ -2 & -4 & -1 \end{pmatrix}$$

- Összeadás **(azonos dimenziók)** $A + B = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$

  (alsó háromszögmátrix) `A + B`
- Transzponálás: $A^T = \begin{pmatrix} 2 & 3 \\ -1 & 5 \\ 0 & 1 \end{pmatrix}$ `A'`
- Szorzás **(megegyező belső dimenziók)**:

  $A * A^T = \begin{pmatrix} 5 & 1 \\ 1 & 35 \end{pmatrix}$ `A*A'`

Műveleti tulajdonságok

$$A = \begin{pmatrix} 2 & -1 & 0 \\ 3 & 5 & 1 \end{pmatrix} \qquad B^T = \begin{pmatrix} -1 & -2 \\ 1 & -4 \\ 0 & 1 \end{pmatrix}$$

- A szorzás **nem** kommutatív (felcserélhető), azaz $A * B^T \neq B^T * A$ (a példa mátrixokkal számolva az eredménynek még a dimenziói sem stimmelnek)
- Igaz az alábbi összefüggés: $(A * B)^T = B^T * A^T$ (Tetszőleges A,B mátrixokra, ahol a megfelelő műveletek értelmezettek)

Speciális mátrixok

- Egységmátrix (3x3) $I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ `eye(3)` (a szorzás egységeleme)
- Diagonális mátrixok $D = \begin{pmatrix} 3 & 0 & 0 \\ 0 & -9 & 0 \\ 0 & 0 & 6 \end{pmatrix}$
  `diag([3,-9,6])` (elemek csak a főátlóban)
- Háromszögmátrixok:
  Alsó: $\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 6 & -9 & 1 \end{pmatrix}$ (lower) Felső: $\begin{pmatrix} 1 & 2 & -7 \\ 0 & 1 & 4 \\ 0 & 0 & 8 \end{pmatrix}$ (upper)

Mátrixok determinánsa

- **Csak négyzetes (n x n) mátrixnak**
- Speciális eset (2x2):

$$A = \begin{matrix} 2 & 3 \\ -1 & 4 \end{matrix}$$

$$det(A) = \begin{vmatrix} 2 & 3 \\ -1 & 4 \end{vmatrix} = (2 * 4) - (-1 * 3) = 8 + 3 = 11$$

```
det(A)
```

- Általában kifejtéssel vagy Vandermonde-determinánssal... lásd DiMat I.

Mátrixok inverze

- Szintén **csak négyzetes mátrixnak**

$$A * A^{-1} = A^{-1} * A = I$$

- Számítás definícióval:

$$A^{-1} = \frac{1}{det(A)} * [\text{adjungált aldeterminánsok}]^T$$

```
inv(A)  A^-1
```

- $\Longrightarrow$ Azon mátrixoknak, melynek a **determinánsa 0**, nincs inverze! (egyedül vannak) $\Longrightarrow$ **szingulárisnak** nevezzük őket
- A többi mátrixnak van inverze, őket nemszingulárisnak nevezzük

Operátorok

Aritmetikai

+	összeadás
-	kivonás
*	szorzás
.*	elemenkénti szorzás
/	"osztás" ($A/B = A * B^{-1}$)
\	"osztás" ($A\backslash B = A^{-1} * B$)
./	elemenkénti osztás
^	hatványozás
.^	elemenkénti hatványozás
'	transzponálás

Logikai

~	nem
&&	és
&	elemenkénti és
||	vagy
|	elemenkénti vagy

Relációk

<	>
<=	>=
==	~=

Mátrixok, vektorok megadása

- $A = \begin{pmatrix} 2 & -1 & 0 \\ 3 & 5 & 1 \end{pmatrix}$ Matlabban: `A = [2 -1 0; 3, 5, 1]`
- A sorokat ; -vel az oszlopokat szóközzel vagy , -vel választjuk el
- Vektorok megadása:
  - A vektor speciális mátrix $\Longrightarrow$ ugyanúgy megadhatjuk, ahogy egy mátrixot
    `a = [1 2 3 4 5 6, 7 8 9 10];` sorvektor
  - Egyszerűsített megadás: `a = 1:10;` szintén sorvektort képez
  - Általánosabban:
    `a = 1:2:7;` $a = (1, 3, 5, 7)$
    `start:lépésköz:end`

Speciális mátrixok megadása

- `A = zeros(2,4);` $A = \begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$
- `A = ones(3);` $A = \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$
- `R = rand(2,3);` $R = \begin{matrix} 0.8147 & 0.1270 & 0.6324 \\ 0.9058 & 0.9134 & 0.0975 \end{matrix}$

  A rand() 0-1 közötti egyenletes eloszlásból generál számokat
- A zeros,ones,rand függvények hívhatók 1 vagy 2 vagy több paraméterrel is: 1 paraméteres esetben négyzetes mátrix az eredmény
- `I = eye(2);` $I = \begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$

Speciális mátrixok megadása

- `D = diag([3 5 1]);` $D = \begin{matrix} 3 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{matrix}$

  Ha az input egy vektor, akkor a kimenet egy diagonális mátrix, ahol a diagonális elemek az eredeti vektor elemei

- `D = [1:3;4:6;7:9];` $D = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{matrix}$

  `d = diag(D);` $\rightarrow$ `[1 5 9]`

  Ha az input egy mátrix, akkor a kimenet egy vektor a diagonális elemekből

Elemhivatkozások

- 1-től indexelünk
- Elemhivatkozás () zárójelekkel

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

```
A = [1:4;5:8;9 10 11 12]
```

```
A(2,3)
```
$\rightarrow 7$

- Az első koordináta a sor, a második az oszlop
- A vektorok egy paraméterrel indexelhetők

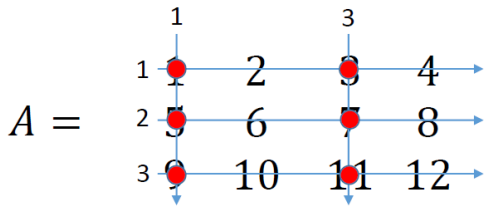
Pl.:
```
a = [1:2:11];
a(3);
```

For hardcore users:

Az fenti mátrix megadható `A = reshape(1:12,4,3)'` módon is. 1:12 előállít egy sorvektort 1-től 12-ig. A reshape utasítás pedig oszlopfolytonosan olvasva azt 4 x 3 -as mátrixba rendezi, amit aztán transzponálunk.

Indexelés vektorokkal

- A mátrixok indexelhetők vektorokkal




```
A([1 2 3],[1 3])
A(1:3,[1,3])
A(1:end,[1 3])
A(:,[1 3])
```

- A kiválasztott sorok és oszlopok metszéspontjából alakul ki az eredmény: részmátrixok

$$ans = \begin{matrix} 1 & 3 \\ 5 & 7 \\ 9 & 11 \end{matrix}$$

Kettőspont operátorok

`j:k`	mint a `[j, j+1, ..., k]`, és üres, ha $j > k$
`j:i:k`	mint a `[j, j+i, j+2i, ..., k]`, és üres, ha $i > 0$ és $j > k$ vagy ha $i < 0$ és $j < k$
`A(:,j)`	A j. oszlopa
`A(i,:)`	A i. sora
`A(:,:)`	a teljes 2D-s tömb. Mátrixoknál ez ugyanaz, mint `A`
`A(j:k)`	eredménye: `A(j),A(j+1),...,A(k)`
`A(:,j:k)`	eredménye: `A(:,j),A(:,j+1),...,A(:,k)`
`A(:,:,k)`	egy 3D-s A tömb k. lapja
`A(i,j,k,:)`	ha A egy 4D-s tömb, az eredményvektor a következőket tartalmazza: `A(i,j,k,1)`, `A(i,j,k,2)`, stb
`A(:)`	A összes eleme egy oszlopban reprezentálva

Gelle Kitti nyomán

Mátrix méretek

- $A = \begin{pmatrix} 2 & -1 & 0 \\ 3 & 5 & 1 \end{pmatrix}$ `A = [2 -1 0; 3 5 1]`
- A mátrix mérete a size() függvénnyel kérhető le `size(A)` egy vektort ad eredményül: $\rightarrow$ `[2 3]`
- Lekérhető speciálisan valamelyik dimenzió mérete is: `size(A,2)` $\rightarrow 3$
- A vektorok hossza a length() függvénnyel kérhető le: `length([1 5 -2 3])` $\rightarrow 4$

Vektorizált műveletek

- Számos művelethez nincs szükség for ciklusra, mert beépített függvények kezelik őket
- Ahelyett, hogy a for ciklus magjában a mátrix minden elemére meghívnánk egy adott függvényt, meghívhatjuk a függvényt az egész mátrixra
- Függvények, melyek elemenként hajtják végre a műveleteket:
  - abs()
  - sqrt()
  - exp()
  - log()
  - sin()
  - cos()
  - tan()
  - cot()
  - round()
  - fix()
  - floor()
  - ceil()
- Statisztika képző függvények:
  - min()
  - max()
  - sum()
  - prod()
  - mean()
  - std()

Példák

`x = 0.7:-0.3:-0.5;` → `[.7 .4 .1 -.2 -.5]`

- Függvények:
  - `abs(x);` → `[.7 .4 .1 .2 .5]`
  - `ceil(x);` → `[1 1 1 0 0]`
  - `floor(x);` → `[0 0 0 -1 -1]`
  - `cos(x);` → `[0.76 0.92 0.995 0.98 0.88]`
  - `mean(x);` → 0.1
- Operátorok

  `a = 1:2:7;` → `[1 3 5 7]`
  - Kivonás: `10 - a;` → `[9 7 5 3]`
  - Számmal való szorzás: `2.*a` → `[2 6 10 14]`
  - Elemenkénti szorzás: `a.*(10-a)` → `[9 21 25 21]`
  - Elemenkénti hatványozás `(10-a).^2` → `[81 49 25 9]`

Fontos megjegyzések

- Az elemenkénti szorzás pont olyan mint a mátrixok összeadása, csak nem összeadunk, hanem szorzunk. $\Longrightarrow$ csak azonos dimenziójú mátrixok szorozhatók össze elemenként
- A sima `*` jel **mátrixszorzást** jelent
- A sima `^` a mátrixszorzás hatványozása. Épp ezért az `a^2` utasítás értelmetlen. Az `a` vektor 1x4-es, két 1x4-es vektor nem szorozható össze (mert a belső dimenziók nem egyeznek). De pl. egy `A` (2x2)-es mátrixra értelmes az `A^3` utasítás

Példa

- $A = \begin{matrix} 1 & 2 \\ 3 & 4 \end{matrix}$ `A = [1 2;3 4]`
- `A^2` $\rightarrow \begin{matrix} 7 & 10 \\ 15 & 22 \end{matrix} \leftarrow$ `A*A`
- `A.^2` $\rightarrow \begin{matrix} 1 & 4 \\ 9 & 16 \end{matrix} \leftarrow$ `A.*A`

Vezérlési szerkezetek

- Szelekciós:

```
if  logikai feltétel
    utasítás(ok)1
[ elseif logikai feltétel2
    utasítás(ok)2 ]
[ else
    utasítás(ok)3 ]
end
```

```
if a>b
  disp('a nagyobb');
elseif b>a
  disp('b nagyobb');
else
  disp('egyenlők');
end
```

- Kezdőfeltételes ismétléses:

```
while feltétel
    utasítások
end
```

Vezérlési szerkezetek

- Számlálásos ismétléses:

```
for ciklusvaltozo = vektor
    utasítások
end
```

```
for i=1:10
 fprintf('%d ',i);
end
```

- Esetkiválasztásos szelekciós:

```
switch kifejezes
    case kifejezes1
        utasítás(ok)1
[    case kifejezes2
        utasítás(ok)2 ] ...
    otherwise
        utasítás(ok)3
end
```

Scriptek, függvények

- Script: a MATLAB parancssorban kiadható utasításokat egy .m kiterjesztésű fájlba gyűjtjük, az így elkészült script-et a fájl nevének parancssorba gépelésével futtatjuk
- Függvény: be és kimeneti paraméterei lehetnek

```
function  [kif1,kif2,...] = foo(be1,be2,...)
    utasítás1;
    ...
    utasításN;
end
```

- A `%` jellel kommenteket írhatunk a kódba

További hasznos utasítások

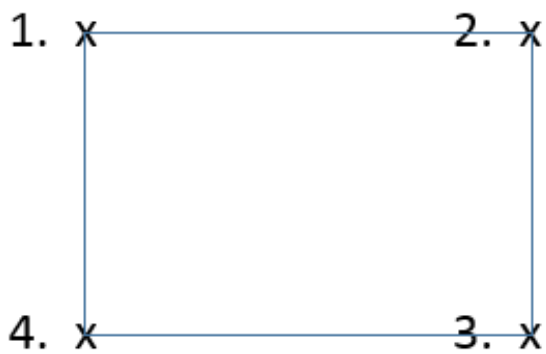
`return`	függvényből visszatérés
`break, continue`	ciklus kényszerített befejezése, léptetése
`nargin,nargout`	be és kimeneti paraméterek száma
`feval`	függvény értékelése az adott helyen
	pl.: `feval('cos',pi);` $\rightarrow$ -1
`input`	érték bekérése konzolról

`fopen,fclose,fprintf,fscanf,disp,sprintf` C-hez hasonlóan

Függvények ábrázolása

- Példa: Ábrázoljuk a cos függvényt a $[-2\pi, 2\pi]$ intervallumon! (a MATLAB-ban beépített konstans a `pi`)
- Vegyünk föl sok pontot az intervallumon:
  `x = -2*pi:0.1:2*pi;`
- Számítsuk ki a cos függvényt: `y = cos(x)`
- Ábrázoljuk: `plot(x,y)`
- Ha ugyenerre az ábrára szeretnénk még rajzolni:
  `hold on;`
- Tegyük rá pirossal a szinus görbét is ugyanezen az intervallumon: `plot(x,sin(x),'r');`
- Tegyük le az ábrát: `hold off;` (a következő plot fölül fogja írni az előző rajzot)

- A `plot` utasítás általánosságban pontokat rajzol ki. Alapértelmezés szerint a megadott sorrendben összeköti a pontokat egy egyenes vonallal. (pont úgy, ahogy általános iskolában az összekötős játékban)
- pl.:



A pontokat sorban összekötjük

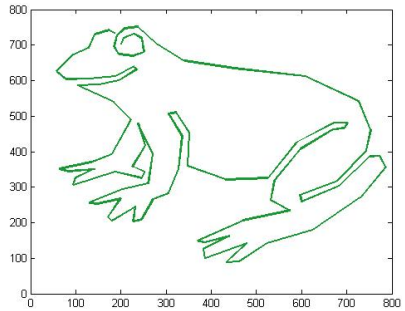
```
X = [1 4 4 1 1];
Y = [3 3 1 1 3];
```

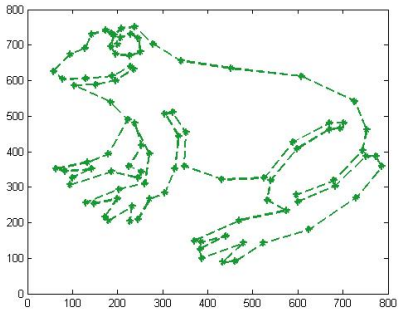
Azért kell 5 pont, hogy zárt legyen

- A `plot(X,Y)` egy kék téglalapot rajzol
- A `plot` számos másfajta paraméterezéssel is hívható (bővebben lásd a matlab dokumentációban)

Plot példa

Megfelelő koordinátákkal az alábbi kép is kirajzolható:





A `plot(x,y,LineSpec)` paraméterezésben a LineSpec egy sztring, ami szabályozza, hogy a pontok hogyan legyenek összekötve. Ha a sztringben szerepel az r(ed), g(reen), b(lue), (blac)k, y(ellow), m(agenta) vagy c(ian) betű, akkor a színt változtatjuk. A . : + * o karakterek a pontok jelét határozzák meg, míg a - : karakterek az összekötő vonal típusát.

`plot(x,y,'g*:')` Zöld színű csillag formájú pontok, szaggatott vonallal összekötve

3Ds ábrázolás

- Az alapvető logika nem változott, pontokat rajzolunk és kötünk össze felületekkel
- plot3, fplot, mesh, surf, ezplot függvények
- Egy egyszerű példa 2D függvények rajzolására:

```
[X,Y] = meshgrid(-5:0.1:5); % alappontok
létrehozása
Z = X.^2-Y.^2;
surf(X,Y,Z);
```

- Egy izgalmasabb felület:

```
load topo;
surf(topo);
```

Gyakorlás 1

1. Hozz létre egy A vektort, amit 0-tól 5-ig tartalmazza a számokat 0.2-es közökkel
2. Transzponáld A-t legyen ez B
3. C oszlopvektorba számold ki B elemeinek négyzetét, D vektorba 10-es alapú logaritmusát, E vektorba természetes alapú logaritmusát
4. Ábrázold egy grafikonon az eddig létrehozott vektorokat folytonos vonallal, illetve az adatpontok jelölésével
5. Készíts egy M mátrixot, melynek első oszlopa A, második oszlopa B, ... ötödik oszlopa E
6. Szorozd össze M-et önmagával elemenként, majd transzponáld a kapott mátrixot és adj hozzá egy megfelelő dimenziójú véletlen mátrixot.
7. A surf() utasítás segítségével ábrázold a mátrixot mint felületet

Gyakorlás 2

1. Hozz létre tetszőleges mátrixokat és demonstrálj rajtuk néhány mátrixtulajdonságot! (nem négyzetes mátrix inverze hibát okoz, csak megfelelő dimenziójúak szorozhatók össze, elemenkénti és mátrixszorzás közötti különbség, mátrixszorzás kommutativitiása stb.)
2. Készíts el egy 10x10-es mátrixot, amely oszlop/sor folytonosan tartalmazza 3-tól 300-ig a 3-mal osztható számokat (for ciklussal vagy reshape-el). Adj meg egy elemhivatkozást (részmátrix képzést), amely elkészíti a 6-tal osztható számok részmátrixát!

Extra feladatok 1.

1. Tölts le az internetről egy kötögetős játékhoz tartozó képet. A kotosJatekVaz.m fájlban írt megjegyzések és vázlat kód alapján készíts el egy kötögetős interaktív játékot!

Extra feladatok 2.

1. Írj egy függvényt az $a * x^2 + b * x + c = 0$ másodfokú egyenlet megoldására! Bemeneti paraméterek: a, b, c. A megoldás tudja kezelni a speciális eseteket is: többszörös megoldás, komplex megoldás.
2. Írj függvényt, ami 2 azonos méretű négyzetes mátrixot vár bemenetként (írjuk ki, ha gond van az inputtal), majd kiszámítja a két mátrix összegét, szorzatát, elemenkénti szorzatát, a két mátrix inverzét, determinánsát!
3. Írj egy függvényt, aminek bemeneti paramétere egy rand(n) mátrix! Hívd meg a cos függvényt százszor egymástól függetlenül a bemeneti mátrixra! A visszatérési érték egy 100 hosszú sorvektor legyen, ami a végrehajtási időket tartalmazza (ld. timeit() függvény)! Számold ki a minimális, maximális és átlagos időtartamot, illetve a mért idők szórását is! Az eredményt az eredmenyek.txt fájlba írja ki a program!

Extra feladatok 3.

1. Használd a későbbiekben a MATLAB-ot és 3D-s megjelenítését, amikor a Kalkulus II. miniZH-kra készülés közben 2 dimenziós függvény analízist kell végezned! (írd be MATLAB-ba a formulát és ellenőrizd, hogy tényleg nyeregpont, minimum, maximum pont van-e az adott helyen)