

# Memento kurzus: Matlab gyakorló feladatok megoldásai

Segédanyag: Molnár Tamás

2019. november 13.

## 1. Feladat

Hozzuk létre a következő  $\mathbf{M}$  mátrixot!

$$\mathbf{M} = \begin{bmatrix} \mathbf{e} & \mathbf{0} & \mathbf{0} & \mathbf{f} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix},$$

ahol  $\mathbf{I}$  és  $\mathbf{0}$  a  $2 \times 2$ -es egység- illetve nullmátrix.

*Megoldás:*

```
M = [e,zeros(2,4),f; eye(6),zeros(6,2)]
```

## 2. Feladat

Az alábbi  $\mathbf{v}_1$  sorvektor,  $\mathbf{v}_2$  oszlopvektor és  $\mathbf{A}$  mátrix segítségével képezzük a  $\mathbf{v}_1 \cdot \mathbf{A} \cdot \mathbf{v}_2$  szorzatot úgy, hogy csak minden harmadik sorban illetve oszlopban levő elemet vesszük figyelembe!

```
v1 = (-10:10);  
v2 = (30:-2:-10)';  
A = (-5:15)' .* (5:25);
```

*Megoldás:*

```
v1(3:3:end)*A(3:3:end,3:3:end)*v2(3:3:end)
```

*Megjegyzés:* Egy oszlopvektort elemenként megszorozva egy sorvektorral mátrixot kapunk eredményül, így építettük fel az  $\mathbf{A}$  mátrixot.

## 3. Feladat

Transzformáljuk a következő mátrixot diagonális alakra sajátvektorai segítségével! Lássuk be, hogy ez megegyezik a sajátértékekből képzett diagonális mátrixszal!

```
M = [-2,7,4  
      5,2,-1  
      1,0,8];
```

*Megoldás:*

```
[T,D]=eig(M);  
T\M*T  
D
```

#### 4. Feladat

Hozzuk létre a  $\text{Fibo}(n)$  függvényt, amely kiszámítja a Fibonacci-sorozat  $n$ . tagját! A sorozat képlete:  $a_1 = 0$ ,  $a_2 = 1$ ,  $a_n = a_{n-1} + a_{n-2}$  ( $n \geq 3$ ,  $n \in \mathbb{N}$ ).

*Megoldás:*

```
% a Fibo.m fajlban
function x=Fibo(n)
if n<3 || (n-floor(n))~=0
    disp('Index negativ vagy nem egesz szam')
else
    a=zeros(1,n);
    a(1)=0; a(2)=1;
    for i=3:n
        a(i)=a(i-1)+a(i-2);
    end
    x=a(end);
end

% eredeti .m fajlban
Fibo(8)
arrayfun(@Fibo,1:10)
```

*Megjegyzés:* Még erre a számításra is létezik beépített Matlab parancs (`fibonacci`). Az általunk írt `Fibo` függvény csak skalár bemenetet képes fogadni. Az `arrayfun` parancs segítségével skalárookra értelmezett függvényt alkalmazhatunk mátrixok esetén is. Ekkor a mátrix minden elemén elvégzi a Matlab a műveletet és az eredményt berakja egy mátrixba.

#### 5. Feladat

Határozzuk meg az alábbi vektor elemeinek négyzetösszegét `for` ciklus használatával és anélkül!

```
v=-20:0.000001:30;
```

*Megoldás:*

```
% 1. megoldas
v*v'
% 2. megoldas
sum(v.*v)
% 3. megoldas
vv=0;
for kk = 1:length(v)
    vv=vv+v(kk)^2;
end
vv
```

*Megjegyzés:* Az egyes megoldások futásidejét a `tic` és `toc` parancsok között futtatva lemérhetjük. Ez alapján láthatjuk, hogy a beépített mátrixműveletek lényegesen gyorsabbak lehetnek, mint a `for` ciklus.

#### 6. Feladat

Ábrázoljuk az  $A \sin t$  időfüggvényt különböző  $A$  amplitúdó értékekre egy közös ábrában!

```
t = 0:2*pi/100:2*pi;
A = 0:2:20;
```

*Megoldás:*

```

% for ciklussal
figure;
hold on;
for kk=1:length(A)
    plot(t,A(kk)*sin(t))
end

% matrix segítségével
figure;
plot(t,A' .*sin(t))

```

## 7. Feladat

Ábrázoljuk az  $f(x) = \int_{-10}^x e^{-t^2} dt$  függvényt az  $x \in [-10, 10]$  tartományon! Az integráláshoz használjuk az `integral` parancsot.

```
x=-10:0.01:10;
```

*Megoldás:*

```

f = @(x)integral(@(t)exp(-t.^2),-10,x);
figure
plot(x,arrayfun(f,x))

```

*Megjegyzés:* Az `integral` parancs mellett numerikus deriválás és integrálás elvégzéshez hasznos lehet a `diff` és a `cumsum` parancs.

```

figure
hold on
plot(x,exp(-x.^2))
plot(x,cumsum(exp(-x.^2)*0.01))
plot(x(2:end),diff(exp(-x.^2)/0.01))

```

## Gyakorló feladat

Készítsünk numerikus szimulációt egy matematikai inga mozgásáról! Az inga mozgását leíró nem-lineáris differenciálegyenlet:

$$\ddot{\varphi}(t) + \sin \varphi(t) = 0, \quad (1)$$

ahol  $\varphi$  az inga függőlegestől mért szöghelyzete radiánban. Az  $\mathbf{y}(t) = [\varphi(t) \ \dot{\varphi}(t)]^T$  állapotvektor bevezetésével a mozgásegyenlet elsőrendű alakba írható:

$$\begin{bmatrix} \dot{y}_1(t) \\ \dot{y}_2(t) \end{bmatrix} = \begin{bmatrix} y_2(t) \\ -\sin y_1(t) \end{bmatrix}. \quad (2)$$

A mozgásegyenlet megoldását számítsuk ki a  $t \in [0, 10\pi]$  időintervallumon a  $\varphi(0) = 30^\circ$ ,  $\dot{\varphi}(0) = 0$  (azaz  $\mathbf{y}(0) = [\pi/6 \ 0]^T$ ) kezdeti feltétellel. A mozgásegyenlet megoldásához használjuk a beépített `ode45` függvényt! Az `ode45` alkalmazásához a sűgőban találunk segítséget (gyorsgombja: F1). Ábrázoljuk a kapott  $\varphi(t)$  függvényt!

*Megoldás:*

```

t0 = 0;
tmax = 10*pi;
phi0 = 30*pi/180;
dotphi0 = 0;
[t,y] = ode45(@(t,y) [y(2);-sin(y(1))], [t0,tmax], [phi0,dotphi0]);

```

```

phi = y(:,1);
figure
plot(t,phi*180/pi)
xlabel('t')
ylabel('\phi')

```

A szimuláció ciklusos ismétlésével vizsgáljuk meg, hogy változik a lengések  $T$  periódusideje a  $\varphi(0)$  kezdeti kitérés függvényében! A szimulációk során legyen  $\varphi(0) = 2^\circ, 4^\circ, \dots, 174^\circ, 176^\circ$ .

*Megoldás:*

```

t0 = 0;
tmax = 10*pi;
phi0 = (2:2:176)*pi/180;
dotphi0 = 0;
opts = odeset('MaxStep',0.1);
T = 0*phi0;
for kk = 1:length(phi0)
    [t,y] = ode45(@(t,y) [y(2);-sin(y(1))], [t0,tmax],...
                 [phi0(kk),dotphi0], opts);
    phi = y(:,1);
    % a felperiodusnal valt eloszor elojelet phi derivaltja
    T(kk) = 2*t(find(diff(phi)>0,1));
end
figure
plot(phi0*180/pi,T/2/pi)
axis([0 180 0 4])
title('Periodusido')
xlabel('\phi_0')
ylabel('T / (2 \pi)')

```

*Megjegyzés:* A  $T = 0*phi0$ ; sorral a  $T$  változót először egy  $phi0$  vektorral egyező méretű nullvektorként vesszük fel. Ez azért hasznos, mert így a  $T$  méretét előre definiáljuk, a  $T$  számára szükséges memóriát lefoglaljuk (egyébként a  $T$  mérete a  $for$  ciklus minden egyes lépésénél növekedne és folyamatosan nőne a memória igény). Az `odeset` paranccsal pedig az `ode45` megoldó beállításait érhetjük el, most a szimulációs időlépés maximális nagyságát állítottuk be, hogy a periódusidőt kellően pontosan láthassuk.