MATLAB OKTATÁS
4. ELŐADÁS
LINEÁRIS, NEMLINEÁRIS ÉS DIFFERENCIÁL-EGYENLETRENDSZEREK MEGOLDÁSA

Dr. Bécsi Tamás
Hegedüs Ferenc

LINEÁRIS EGYENLETRENDSZEREK MEGOLDÁSA

- A MATLAB a lineáris egyenletrendszerek megoldásához külön operátorral rendelkezik.
- Az **`mldivide`**, azaz **`\`** operátor az $\boldsymbol{Ax} = \boldsymbol{B}$ lineáris egyenlet-rendszer $\boldsymbol{x}$ megoldását szolgáltatja.
- Szintaktika:

```
x = A \ B
x = mldivide(A, B)
```

- Működés:
  - Ha $\boldsymbol{A}^{\boldsymbol{n}\times\boldsymbol{n}}$ négyzetes mátrix és $\boldsymbol{B}^{\boldsymbol{n}\times\boldsymbol{k}}$ mátrix, akkor $\boldsymbol{x}$ az $\boldsymbol{Ax} = \boldsymbol{B}$ lineáris egyenletrendszer megoldása.
  - Ha $\boldsymbol{A}^{\boldsymbol{m}\times\boldsymbol{n}}$ mátrix, ahol $\boldsymbol{m} \neq \boldsymbol{n}$, és $\boldsymbol{B}^{\boldsymbol{m}\times\boldsymbol{k}}$ mátrix, akkor $\boldsymbol{x}$ az $\boldsymbol{Ax} = \boldsymbol{B}$ alulhatározott lineáris egyenletrendszer legkisebb négyzetek szerinti megoldása.

LINEÁRIS EGYENLETRENDSZEREK MEGOLDÁSA PÉLDA

```
%% Ordinary square-size problem
% Solve A.x = b
A = [1, 2;
     2, 1];
b = [3;
     3];
x = A \ b;
disp(x);
%% Underdetermined system
% Obtain least-square solution for A.x = b
A = [2, 4, 0;
     0, 5, 10];
b = [ 8;
     20];
x = A \ b;
disp(x);
```

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 4 & 0 \\ 0 & 5 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 8 \\ 20 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} t \\ 2 - \frac{t}{2} \\ 1 + \frac{t}{4} \end{bmatrix}$$

POLINOM GYÖKEINEK MEGHATÁROZÁSA

- Polinomok gyökkereséséhez a **`roots`** függvény használható. A gyökök megadásával a polinom együtthatóinak számítása a **`poly`** függvény segítségével történhet.
- Szintaktika:

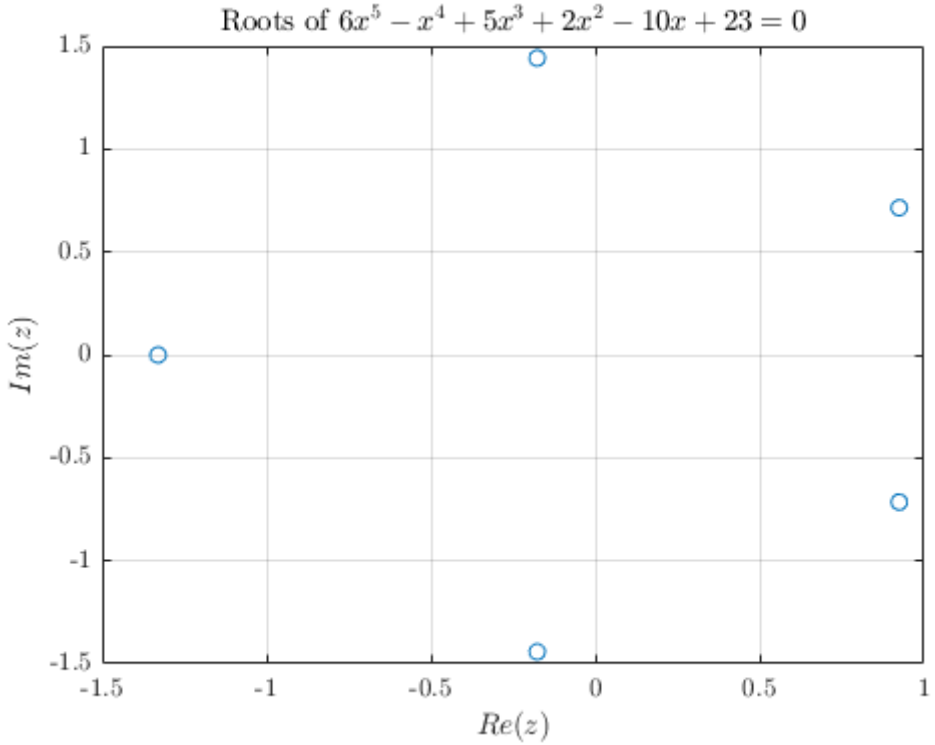
```
r = roots(p)
p = poly(r)
```

- Működés:
  - A **`roots`** függvény a **`p`** együtthatóvektorral megadott polinom gyökeit számolja ki. A p vektor $n + 1$ polinom együtthatót tartalmaz, x^n együtthatójától kezdve, és visszatér az ezeket tartalmazó vektorral.
    Pl.: **`p = [2 0 7 1]`** a $2x^3 + 7x + 1$ polinomot reprezentálja.
  - A **`poly`** függvény az **`r`** gyökökkel adott polinom **`p`** együtthatóvektorát számítja ki, és visszatér vele.

POLINOM GYÖKEINEK MEGHATÁROZÁSA PÉLDA

solve/polyn.m
solve/poly_plot.m

```
%% Find roots
% Specify polynomial 6.x^5 - x^4 + 5.x^3 +
% 2.x^2 - 10.x + 23 = 0
p = [6, -1, 5, 2, -10, 23];
r = roots(p);
disp(r);
% Re-create coefficient vector based on
roots
p2 = poly(r);
disp(p2);
% Evaluate polynomial at roots
y = polyval(p, r);
disp(y);
%% Plot roots
plot(r, 'o');
```



POLINOMOK SZORZÁSA, OSZTÁSA

- Polinomok szorzásához és osztásához a **conv** és **deconv** függvények használhatóak.
- Szintaktika:

```
m = conv(p, s)
[q, r] = deconv(p, s)
```

- Működés:
  - A **conv** függvény a **p** és **s** együtthatóvektorokkal megadott polinomok szorzatát számítja ki, és visszatér a szorzat polinom együtthatóvektorával (**m**).
  - A **deconv** függvény a **p** és **s** együtthatóvektorokkal megadott polinomokkal polinomosztást hajt végre, és a hányados polinomjának (**q**) és a maradék polinomnak (**r**) az együtthatóvektoraival tér vissza.

FÜGGVÉNY „MUTATÓK” (FUNCTION HANDLES)

- A függvény mutatók (function handles) olyan változók, melyeket egy függvényhez társítunk.
- Lehetővé teszik, hogy:
  - Rajtuk keresztül függvényhívásokat valósítsunk meg.
  - Függvényt adjunk át argumentumként másik függvénynek.
  - Paraméterezzünk olyan függvényeket, melyeknek kötelező argumentum-listája van.
  - Külön fájlban nem létező, ún. névtelen függvényeket (anonymous functions) hozzunk létre.

FÜGGVÉNY „MUTATÓK” (FUNCTION HANDLES) PÉLDA

fcn_handle.m
eval_fcn.m

```
%% Create function handles
% Create function handle for a second order
% polynomial.
poly2 = @ (x, a, b, c) a * x^2 + b * x + c;
% Function handle with fixed a, b, c
% parameters.
poly2_111 = @(x) poly2(x, 1, 1, 1);
%% Display results
% Call the function handle with free
% parameters.
disp(poly2(2, 1, 1, 1));
% Call the function handle with fixed
% parameters.
disp(poly2_111(2));
% Pass the function handle to another
% function to evaluate.
disp(eval_fcn(poly2_111, 2));
```

```
% Function that evaluates another
% function passed by the function
% handle parameter fcn, at x.
function y = eval_fcn(fcn, x)
    y = fcn(x);
end
```

NEMLINEÁRIS EGYENLETRENDSZEREK MEGOLDÁSA

- Nemlineáris egyenletrendszerek megoldására az Optimization Toolbox **`fsolve`** függvénye használható.
- Az **`fsolve`** függvény az $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ nemlineáris egyenletrendszer $\boldsymbol{x}$ megoldását szolgáltatja, ahol $\boldsymbol{x}$ vektor vagy mátrix, $\boldsymbol{F}(\boldsymbol{x})$ pedig vektorértékű függvény.
- Szintaktika:

```
[x_s, f_s] = fsolve(F, x_0, opts)
```

- Működés:
  - Az **`fsolve`** az **`x_0`** pontból kiindulva megpróbálja az **`F`** függvény mutatóval megadott $F(x) = 0$ nemlineáris egyenletrendszer megoldását megtalálni, az **`opts`** argumentumban megadott beállítások mellett.
  - A visszatérési értékek a megoldás helye **`x_s`** és értéke **`f_s`** ($f_s = F(x_s)$).
  - Az **`opts`** argumentumban átadott beállítás struktúra az **`optimoptions`** függvény segítségével készíthető el.

FSOLVE BEÁLLÍTÁSOK (OPTIMOPTIONS)

Név	Érték
Algorithm (algoritmus)	’trust-region-dogleg’ (*) ’trust-region’ ’levenberg-marquardt’
Display (kijelzés)	’off’ or ’none’ nincs ’iter’ információ minden iterációs lépésben ’final’ (*) csak végleges információ
FunctionTolerance (függvényérték tűrés)	Kilépési tűrés a függvényértékre, pozitív skalár. Az alapértelmezett érték 1e-6.
StepTolerance (lépés méret tűrés)	Kilépési tűrés a lépés méretére, pozitív skalár. Az alapértelmezett érték 1e-6.
OptimalityTolerance (optimalitás tűrés)	Kilépési tűrés az elsőrendű optimalitásra, pozitív skalár. Az alapértelmezett érték 1e-6.
PlotFcn (grafikus megjelenítő függvény)	@optimplotx aktuális pont ábrázolása @optimplotfval függvényérték ábrázolása @optimplotstepsize aktuális lépés méret ábrázolása

* Alapértelmezett opció.

NEMLINEÁRIS EGYENLETRENDSZEREK MEGOLDÁSA PÉLDA

solve/mf.m
solve/nonlin.m
solve/nonlin_plot.m

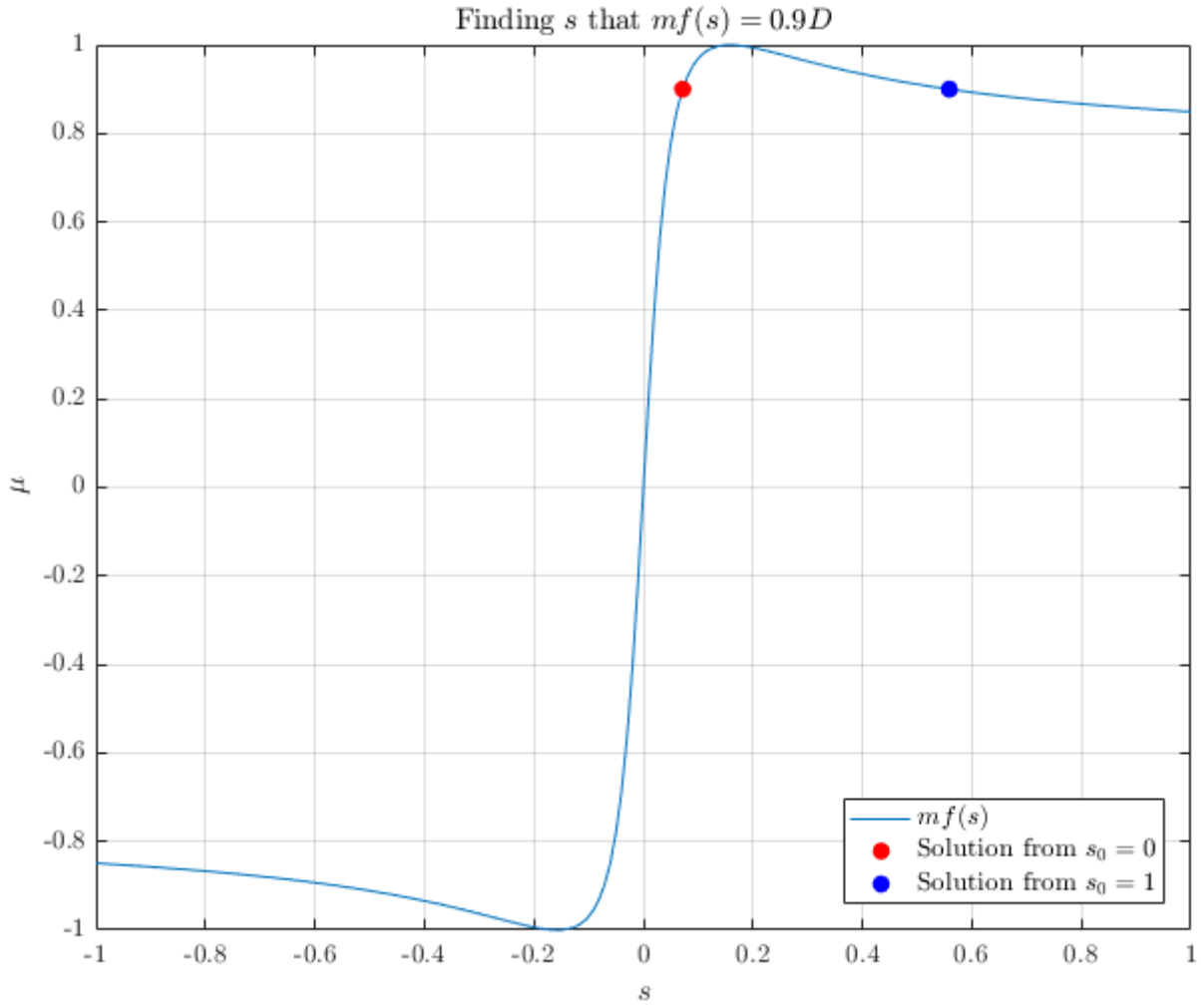
```
%% Parameters of Pacejka Magic formula
D = 1.0;
C = 1.45;
B = 15.0;
E = 0.4;
%% Compute slip curve
% Signed slip values are valid from -1 to 1
s = -1:0.01:1;
% Calculate adhesion coefficient curve
mu = mf(s, D, C, B, E);
%% Solve nonlinear equation to find the slip
%  value where mu is 0.9 * mu_max
mu_max90 = 0.9 * D;
% Function handle that represents F(x) = 0
fcn = @(x) mf(x, D, C, B, E) - mu_max90;
% Solve equation from initial point 0
[s_max90_0, fcn_max90_0] = fsolve(fcn, 0);
% Solve equation from initial point 1
[s_max90_1, fcn_max90_1] = fsolve(fcn, 1);
```

```
% Display result
disp(s_max90_0);
disp(s_max90_1);
%% Plot results
figure();
hold on;
% Plot slip curve
plot(s, mu);
% Plot points of solution
plot(s_max90_0, mu_max90, 'o');
plot(s_max90_1, mu_max90, 'o');
```

```
% Hans Pacejka's Magic Formula
% tire model equation
function mu = mf(s, D, C, B, E)
    mu = D .* sin(C .* atan(B .* s...
        - E .* (B .* s - atan(B .* s))));
end
```

EFOP-3.6.3-VEKOP-16-2017-00001

NEMLINEÁRIS EGYENLETRENDSZEREK MEGOLDÁSA PÉLDA



KÖZÖNSÉGES DIFFERENCIÁLEGYENLET-RENDSZEREK MEGOLDÁSA

- A MATLAB számos függvényt kínál *elsőrendű közönséges differenciálegyenletek kezdeti érték problémáinak* megoldására. Ezek az **ode45**, **ode23**, **ode113**, **ode15s**, **ode23s**, **ode23t**, **ode23tb**, és az **ode15i**.
- Mindegyik megoldó bizonyos típusú problémákra használható a leghatékonyabban.
- Merev (stiff) differenciálegyenlet-rendszernek nevezünk olyan differenciálegyenlet-rendszereket, melyeket az explicit módszerek csak elfogadhatatlanul kicsi időlépés mellett képesek megoldani.
- Általános, nem merev (nonstiff) differenciálegyenlet-rendszerek megoldására az **ode23** és az **ode45**, míg merev (stiff) differenciálegyenlet-rendszerek megoldására az **ode15s** és az **ode23s** használható jól.

EXPLICIT NUMERIKUS KDE MEGOLDÁS

- Adott az alábbi kezdeti érték probléma: $\dot{y}(t) = f\big(t, y(t)\big)$, $y(t_0) = y_0$
- Explicit Euler-módszer (előrehaladó):
  - A differenciál-hányados az alábbi előrehaladó differencia-hányadossal közelíthető:

    $$\dot{y}(t) = f\big(t, y(t)\big) \approx \frac{y(t+h)-y(t)}{h}$$

  - Így a megoldás a $t + h$ időpillanatban az alábbiak szerint közelíthető:

    $$y(t + h) \approx y(t) + hf\big(t, y(t)\big)$$

  - Tegyük fel, hogy a kezdetiérték-probléma megoldása a $[0; t_F]$ intervallumon érdekes számunkra. Osszuk fel az intervallumot ekvidisztáns módon N intervallumra:

    $$t_i = ih,\ i = 0 \ldots N \Rightarrow h = \frac{t_F}{N}$$

  - A t_{i+1} időpillanatban a megoldás az alábbi rekurzív formulával közelíthető, mely – mivel az előző időpillanatbeli értékek ismertek – azonnal kiértékelhető:

    $$y_{i+1} = y_i + hf(t_i, y_i)$$

IMPLICIT NUMERIKUS KDE MEGOLDÁS

- Adott az alábbi kezdeti érték probléma: $\dot{y}(t) = f\big(t, y(t)\big)$, $y(t_0) = y_0$
- Implicit Euler-módszer (hátrahaladó):
  - A differenciál-hányados az alábbi hátrahaladó differencia-hányadossal közelíthető:

    $$\dot{y}(t) = f\big(t, y(t)\big) \approx \frac{y(t) - y(t-h)}{h}$$

  - Így a megoldás a $t + h$ időpillanatban az alábbiak szerint közelíthető:

    $$y(t) \approx y(t-h) + hf\big(t, y(t)\big)$$

  - Tegyük fel, hogy a kezdetiérték-probléma megoldása a $[0; t_F]$ intervallumon érdekes számunkra. Osszuk fel az intervallumot ekvidisztáns módon N intervallumra:

    $$t_i = ih,\ i = 0 \ldots N \Rightarrow h = \frac{t_F}{N}$$

  - A t_{i+1} időpillanatban a megoldás az alábbi rekurzív formulával közelíthető:

    $$y_{i+1} = y_i + hf(t_{i+1}, y_{i+1})$$

  - A fenti formula nem értékelhető ki azonnal, egy egyenletrendszert kell numerikusan megoldani, hogy y_{i+1} értékét megkapjuk.

MATLAB ODE

- A MATLAB **ode** függvényei az $\dot{\boldsymbol{y}}(\boldsymbol{t}) = \boldsymbol{f}\big(\boldsymbol{t}, \boldsymbol{y}(\boldsymbol{t})\big)$, $\boldsymbol{y}(\boldsymbol{t_0}) = \boldsymbol{y_0}$ kezdeti érték probléma megoldását szolgáltatják a megadott $\boldsymbol{t_s} = [\boldsymbol{t_0}, \boldsymbol{t_f}]$ intevallumon.
- Szintaktika:

[t, y] = odeXXX(f, t_s, y_0, opts)

- Működés:
  - Az **`odeXXX`** az **`y_0`** kezdeti értékből kiindulva a **`t_s = [t_0, t_f]`** intervallumon megoldja az **`f`** függvény mutatóval megadott $\dot{y}(t) = f\big(t, y(t)\big)$ differenciálegyenlet-rendszert az **`opts`** argumentumban megadott beállítások mellett.
  - A visszatérési értékek a kiszámított megoldás-értékek **`y`** és a hozzájuk tartozó időpontok **`t`** oszlopvektorai.
  - Az **`opts`** argumentumban átadott beállítás struktúra az **`odeset`** függvény segítségével készíthető el.

ODE BEÁLLÍTÁSOK (ODESET)

Név	Érték
RelTol (relatív tűrés)	Megengedhető hiba relatív értéke minden egyes megoldás-komponens esetén, pozitív skalár.
AbsTol (abszolút tűrés)	Megengedhető hiba abszolút értéke. Pozitív skalár érték esetén minden egyes megoldás komponensre vonatkozik. A komponensekre egyenként is megfogalmazható, ekkor a komponensek számával megegyező hosszúságú vektor.
OutputFcn (kimeneti grafikus megjelenítő függvény)	@odeplot minden megoldáskomponens ábrázolása időben @odeprint megoldás és időlépés ábrázolása
InitialStep (kezdeti időlépés)	A megoldónak javasolt kezdeti időlépés.
MaxStep (maximális időlépés)	A megoldó által használható maximális időlépés.

MATLAB ODE PÉLDA

ode/ode.m
ode/ode_plot.m
ode/vdp.m
ode/vdp_plot.m
ode/vdp100.m

```
%% Inputs
% Specify time span
t_0 = 0;
t_f = 10;
% Specify initial value y(t0)
y_t_0 = 10;
%% Solve dy(t) = t, y(0) = 10
% Function that evaluates dy(t) = t
f_1 = @(t, y) t;
% Solve the differential equation
[t_1, y_1] = ode45(f_1, [t_0, t_f], y_t_0);
%% Solve dy(t) = y(t), y(0) = 10
% Function that evaluates dy(t) = y
f_2 = @(t, y) y;
% Solve the differential equation
[t_2, y_2] = ode45(f_2, [t_0, t_f], y_t_0);
```

```
%% Plot results
figure;
subplot(2, 1, 1)
% Plot solution of dy(t) = t
plot(t_1, y_1);
subplot(2, 1, 2)
% Plot solution of dy(t) = t
plot(t_2, y_2);
```

$$\dot{y}(t) = t, \qquad y(0) = 10$$

$$y(t) = \frac{t^2}{2} + 10$$

$$\dot{y}(t) = y(t), \qquad y(0) = 10$$

$$y(t) = 10e^t$$