16. DIFFERENCIÁLEGYENLETEK - PEREMÉRTÉK FELADATOK

Eddig a közönséges differenciálegyenletek kezdeti érték feladatával foglalkoztunk, amikor a függvény és deriváltjainak értéke a kérdéses intervallum kezdőpontjában volt megadva. Elsőrendű esetben a függvény értéke a kezdőpontban, másodrendű esetben a függvény és első deriváltjának az értéke a kezdőpontban, harmadrendű esetben a függvény, az első és a második derivált értéke a kezdőpontban stb. Peremérték feladat esetében a függvény és deriváltjainak értékei közül legalább az egyik nem a kezdőpontban, hanem a végpontban adott, és így kell megkeressük azt a függvényt, ami kielégíti a feltételeket.

Nézzünk egy másodrendű közönséges differenciálegyenletet az [a,b] intervallumon:

$$\frac{d^2y}{dt^2} = f\left(t, y, \frac{dy}{dt}\right)$$

A másodrendű differenciálegyenlet megoldásához két feltétel szükséges. Kezdeti érték feladat esetében ez a függvény (y) és az első deriváltjának ($\frac{dy}{dt}$) az értéke a kezdőpontban. Peremérték feladat esetében több változat lehetséges. Lehet adott a függvényérték a kezdő és a végpontban, ezt Dirichlet-peremfeltételnek nevezzük:

$$y(a) = Y_a, \qquad y(b) = Y_b$$

Lehet adott az első derivált értéke a két végpontban, ezt Neumann-peremfeltételnek nevezzük:

$$\left.\frac{dy}{dt}\right|_{t=a} = D_a; \qquad \left.\frac{dy}{dt}\right|_{t=b} = D_b$$

Vagy lehetnek vegyesen pl. a függvény értéke a kezdőpontban és az első derivált értéke a végpontban. Vizsgáljuk azokat az eseteket általánosan, amikor a probléma visszavezethető elsőrendű explicit differenciálegyenlet rendszerre. Ekkor a megoldandó n-ed rendű differenciálegyenlet:

$$\frac{d^ny}{dt^n} = f\left(t, y, \frac{dy}{dt}, \frac{d^2y}{dt^2}, \cdots, \frac{d^{n-1}y}{dt^{n-1}}\right)$$

Vezessünk be ismét egy w vektorváltozót a függvényre és deriváltjaira a jobb oldalon:

$$w = \left(y, \frac{dy}{dt}, \frac{d^2y}{dt^2}, \cdots, \frac{d^{n-1}y}{dt^{n-1}}\right)$$

Ekkor a megoldandó elsőrendű differenciál egyenletrendszer a következő lesz:

$$\frac{d\boldsymbol{w}}{dt} = \boldsymbol{f}(t, \boldsymbol{w})$$

A két peremen ($t = a$ és $t = b$) ismertek a következő értékek:

$$\boldsymbol{w}_i(a) = \boldsymbol{A}_i; \qquad i = 1, \cdots, k$$

$$\boldsymbol{w}_j(b) = \boldsymbol{B}_j; \qquad j = k+1, \cdots, n$$

Vagyis az i indexű elemek a kezdőpontban ($\boldsymbol{A}_i$), a j indexű elemek a végpontban ismertek ($\boldsymbol{B}_j$).

TÜZÉRSÉGI MÓDSZER (SHOOTING METHOD)

A megoldás egyik lehetséges módja, hogy visszavezetjük a feladatot egy kezdeti érték probléma ismételt megoldására, ezt nevezzük **tüzérségi módszernek**.

A megoldás elve az lesz, hogy megoldjuk a kezdeti érték problémát egy felvett kezdeti $\boldsymbol{w}_j(a) = \boldsymbol{u}_j$ értékre, majd ellenőrizzük a $\boldsymbol{w}_j(b) = \boldsymbol{B}_j$ végfeltétel fennállását (a differenciálegyenlet rendszer numerikus megoldásával). Ha eltérés van, akkor módosítjuk az *u* értékét. A felvett kezdeti $\boldsymbol{u}_j$ és a végpontbeli $\boldsymbol{w}_j(b)$ értékek közötti kapcsolatot egy $\boldsymbol{g}(u)$ függvénnyel adhatjuk meg:

$$\boldsymbol{w}_j(b) = \boldsymbol{g}_j(u)$$

Ennek kell egyenlőnek lennie a végpontban ismert értékekkel, $\boldsymbol{B}_j$-vel, vagyis:

$$\boldsymbol{g}_j(u) = \boldsymbol{B}_j$$

Az ismeretlen kezdeti értékeket az alábbi (*n-k*) változós függvények zérus helyeinek megkeresésével határozhatjuk meg:

$$\boldsymbol{h}_j(\boldsymbol{u}) = \boldsymbol{g}_j(\boldsymbol{u}) - \boldsymbol{B}_j = \boldsymbol{0}$$

TÜZÉRSÉGI MÓDSZER ALKALMAZÁSA

Fellövünk egy petárdát függőlegesen felfelé, a petárda 5 másodperc elteltével robban fel. A petárda függőleges mozgását, ha eltekintünk a légellenállástól, akkor a következő másodrendű differenciálegyenlettel írhatjuk le:

$$\frac{d^2y}{dt^2} = -g$$

1) Mekkora sebességgel kell fellőnünk a petárdát, ha azt szeretnénk elérni, hogy pontosan 40 méter magasan robbanjon fel? (Azaz 5 másodperc múlva pontosan 40 méter magasan legyen.)
2) Mekkora lesz a petárda által elért maximális magasság?
3) Hány másodperc után lesz pontosan 35 m magasan?

(Az utolsó két kérdés a korábbiakban tanultak gyakorlása, megoldásuk csak a peremérték feladat megoldása után lehetséges.)

A peremfeltételek tehát:

$$y(0) = 0; \qquad y(5) = 40;$$

A kezdeti érték feladatoknál tanult módon alakítsuk át másodrendű differenciálegyenletet $\left(\frac{d^2y}{dt^2} = f\left(t, y, \frac{dy}{dt}\right)\right)$ két elsőrendű differenciálegyenletből álló rendszerré! Ehhez vezessünk be egy kételemű vektorváltozót, a függő változókra, az első elem a petárda függőleges helyzete (*y*) legyen, a második elem pedig az első derivált, vagyis a petárda függőleges sebessége ($\frac{dy}{dt}$):

$$w = \begin{pmatrix} y & \frac{dy}{dt} \end{pmatrix}$$

Ekkor $w_1 = y$ (függőleges pozíció) és $w_2 = \frac{dy}{dt}$ (függőleges sebesség). Az elsőrendű differenciálegyenlet rendszert az új vektorváltozó elemeinek első deriváltjai adják:

$$f_1 = \frac{dw_1}{dt} = \frac{dy}{dt} = w_2$$

$$f_2 = \frac{dw_2}{dt} = \frac{d^2y}{dt^2} = -g$$

1) Mekkora sebességgel kell fellőnünk a petárdát, ha azt szeretnénk elérni, hogy pontosan 40 méter magasan robbanjon fel? (Azaz 5 másodperc múlva pontosan 40 méter magasan legyen.)

Először oldjuk meg Runge-Kutta módszerrel úgy az egyenletrendszert, hogy felveszünk különböző kezdeti értékeket a sebességre. Legyen $w_2(0) = 20{,}30{,}40{,}50\ ^m/_s$! $\boldsymbol{w}$ egy vektor: $\boldsymbol{w} = [w_1, w_2]$, ahol w_1 a függőleges elmozdulás érték, w_2 pedig az első derivált, vagyis a sebesség értéke.

A differenciálegyenlet rendszert megadhatjuk külön fájlban, pl. a diff_petarda.m fájlban:

```
> function F = diff_petarda(t,w)
>   g = 9.81;
>   f1 = w(2);
>   f2 = -g;
>   F = [f1; f2];
> end
```

Vagy megadhatjuk egysoros függvényként is abban a fájlban ahová dolgozunk (pl. petarda_felloves.m):

```
> g = 9.81;
> dwdt = @(t,w) [w(2); -g]
```

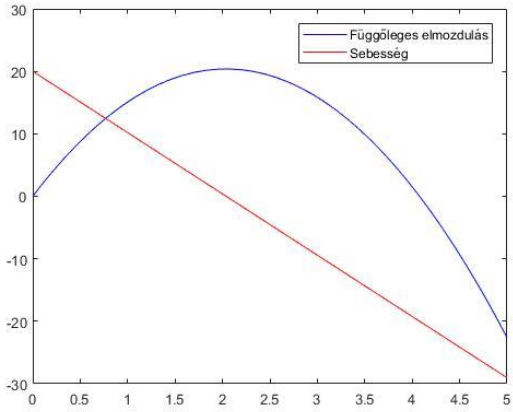
A Runge-Kutta módszerhez használjuk a Matlab beépített **ode45** parancsát! A differenciál egyenlet rendszert külön fájlban definiálva a függvény neve elé kell tenni egy @ szimbólumot. Először adjunk meg 20 m/s kezdeti sebességet, és $0 \leq t \leq 5\ s$ intervallumot!

```
> v0 = 20;
> [T W] = ode45(@diff_petarda,[0; 5],[0; v0]); % külön fájlban
> % vagy
> [T W] = ode45(dwdt,[0; 5],[0; v0]); % egysoros függvényként
```

Az eredményeknél T vektorban a lépésközök vannak a [0, 5] intervallumon. W egy mátrix két oszloppal és annyi sorral, amennyi elem a T vektorban van. W első oszlopában a függőleges elmozdulás értékek vannak (y), a második oszlopban pedig az első deriváltak $\left(\frac{dy}{dt}\right)$, vagyis a sebességek. Ábrázoljuk az elmozdulásokat és a sebességeket az idő függvényében!

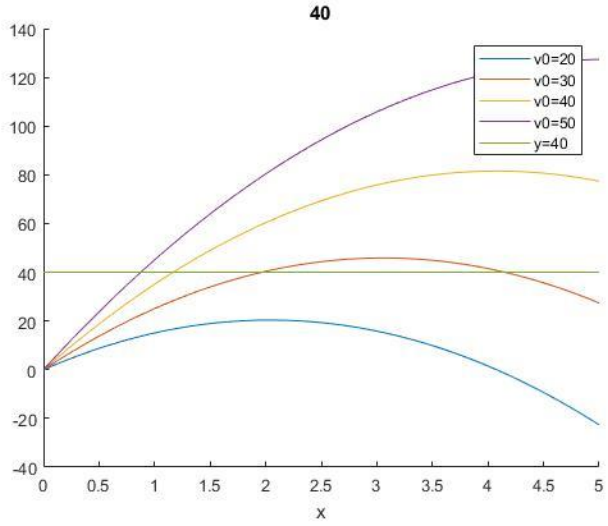
```
> figure(1);
> Y = W(:,1); % magasság
> V = W(:,2); % függőleges sebesség
> plot(T,Y,'b',T,V,'r')
> legend('Függőleges elmozdulás','Sebesség')
> Y(end) % -22.6250
```

Az ábrából láthatjuk, hogy 20 m/s kezdősebességgel nem is közelítjük meg a 40 m-es magasságot. Ezzel a kezdeti sebességgel az intervallum végén negatív értéket kapunk a pozícióra, tehát már rég leesik a földre, mire felrobban.



Vegyünk fel több kezdeti értéket (20,30,40,50 m/s) és most csak a függőleges helyzetet jelenítsük meg! Írassuk ki a végpontbeli magasság értékeket is (W első oszlopának az utolsó eleme)!

```
figure(2); hold on;
for vi=20:10:50
   [T W] = ode45(dwdt,[0; 5],[0; vi]);
   Y = W(:,1); % magasság
   fprintf('v0=%d m/s; y(5)=%f m\r\n',vi,Y(end))
   plot(T,Y);
end
plot([0,5],[40,40]);
legend('v0=20','v0=30','v0=40','v0=50','y=40')
```




```
v0=20 m/s; y(5)=-22.625000 m
v0=30 m/s; y(5)=27.375000 m
v0=40 m/s; y(5)=77.375000 m
v0=50 m/s; y(5)=127.375000 m
```

Az ábrából és az eredményekből is láthatjuk, hogy az 5 másodperces intervallum végén a 30 m/s kezdősebességgel kevesebb, a 40 m/s kezdősebességgel több mint 40 m-es magasságba jutottunk. Valahol a kettő között lesz a megoldás.

Oldjuk meg a feladatot a tűzérségi módszerrel!

Az ismeretlen kezdeti feltétel, vagyis most a kezdeti sebesség, legyen u, és írjuk fel ennek egy g függvényében a végpontbeli függvény értéket, és vizsgáljuk meg, hogy ez mikor lesz 40!

$$g(u) = 40$$

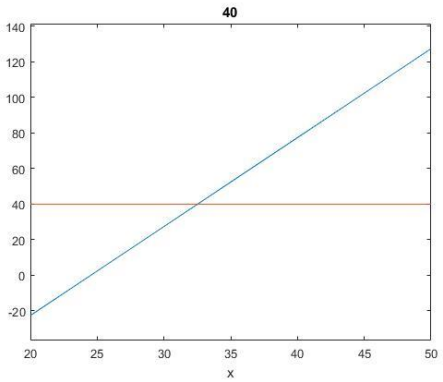
Vagyis keressük az $h = g(u) - 40$ zérushelyét!

Írjuk meg először egy külön fájlban a g függvényt, ami a kezdőérték függvényében visszaadja a végpontbeli magasság értéket! Ehhez a 'diff_petarda.m' külön fájlban megírt differenciálegyenlet rendszert használhatjuk. Legyen ez a fájl a **petarda_magassag.m** fájl!

```
> function y = petarda_magassag(u)
>  [T W] = ode45(@diff_petarda,[0; 5],[0; u]);
>  Y = W(:,1); % magasságok
>  y = Y(end) ); % magasság az intervallum végén
> end
```

Ábrázoljuk a kezdősebesség függvényében az 5 másodpercnél elért magasságokat 20 és 50 m/s kezdősebességek között!

```
> figure(3)
> fplot(@petarda_magassag,[20, 50]);
> hold on; plot([20,50],[40,40]);
```

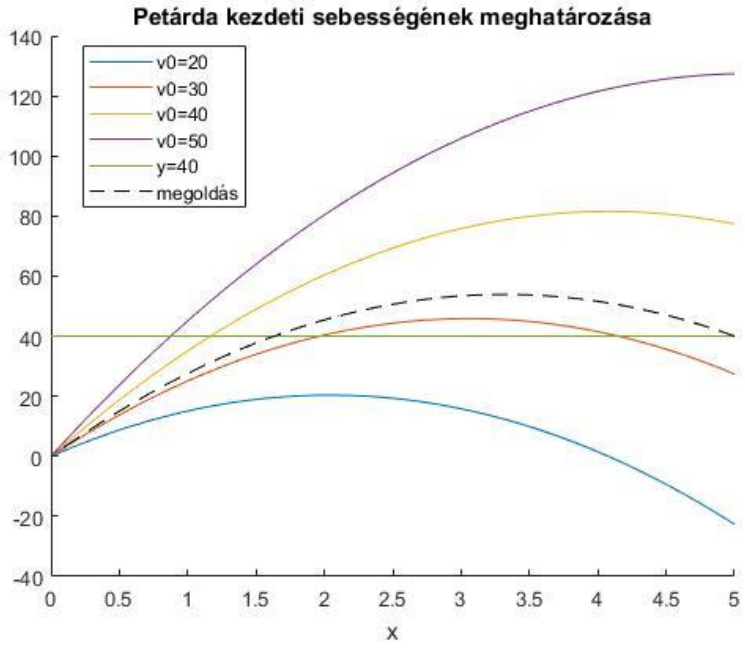


Az ábra alapján valahol 30 és 35 m/s között kell felvennünk a kezdőértéket a sebességhez. Legyen most $u_0 = 32$!

```
> % a hiányzó kezdőérték meghatározás
> h = @(u) petarda_magassag(u)-40
> v0 = fzero(h,32) % 32.5250
```

Rajzoljuk fel a megoldást szaggatott vonallal a többi kezdőérték mellé az ábrába!

```
> [T W] = ode45(@diff_petarda,[ta; tb],[y0; v0]);
> figure(2);
> plot(T,W(:,1),'k--');
> legend('v0=20','v0=30','v0=40','v0=50','y=40','40 m',
  'megoldás','Location','best')
> title('Petárda kezdeti sebességének meghatározása')
```



2) Mekkora lesz a petárda által elért maximális magasság?

Miután megtaláltuk a keresett függvényt a differenciálegyenlet megoldásával, egy egyszerű szélsőérték kereséssel erre a kérdésre is könnyen megfelelhetünk, felhasználva a korábbiakban tanultakat! Ehhez illesszünk egy spline-t a függvény kiszámolt pontjaira, majd alakítsuk át a feladatot minimumkereséssé, a függvény minusz egyszeresének definiálásával!

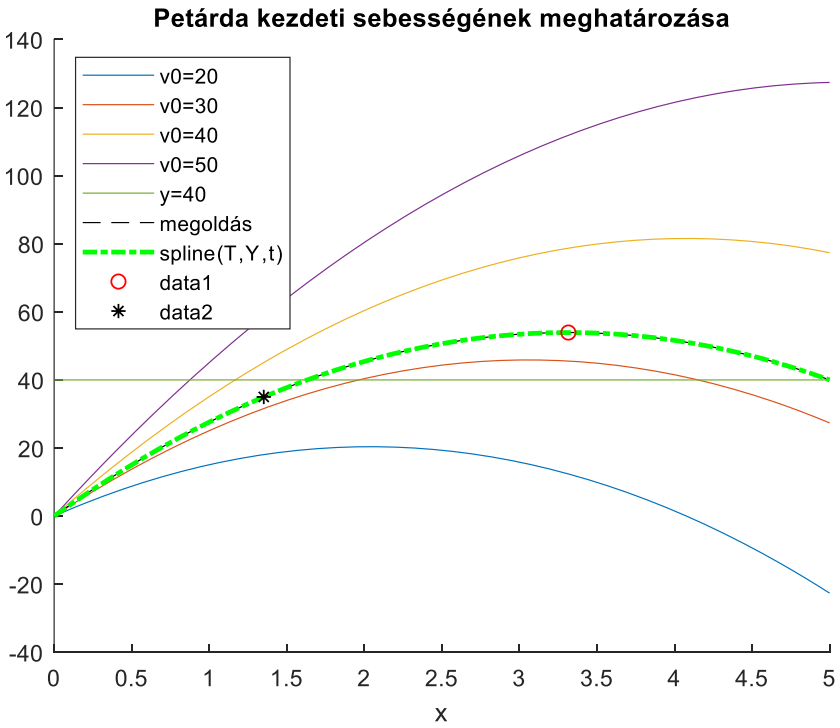
```
> % Elért maximális magasság?
> sp = @(t) spline(T,Y,t) % spline illesztés
> fplot(sp,[0,5],'g-.','LineWidth',2)
> % minimum keresési feladattá alakítás
> spmin = @(t) -1*sp(t)
> % kezdőérték az ábrából: 3.5 s
> tmax = fminsearch(spmin,3.5) % 3.3155
> ymax = sp(tmax) % 53.9182
> plot(tmax,ymax,'ro')
```

Tehát az elért maximális magasság: 53.918 m.

3) Hány másodperc után lesz pontosan 35 m magasan?

Ezt a feladatot nemlineáris egyenlet gyökhelykeresésére vezethetjük vissza. Meg kell oldanunk a következő egyenletet: $y(t) = 35$. Felhasználhatjuk az előző feladatban illesztett spline-t, és deiniáljuk a megoldandó függvényt 0-ra rendezve.

```
> % Hány másodperc után lesz 35 méter magasan a petárda?
> f35 = @(t) sp(t)-35
> % kezdőérték az ábrából: 1.5
> t35 = fzero(f35,1.5) % 1.3516
> % ell.
> y35 = sp(t35) % 35
> plot(t35,y35,'k*')
```



PEREMÉRTÉK FELADAT MEGOLDÁSA A MATLAB BEÉPÍTETT FÜGGVÉNYÉVEL

Sok más módszer is létezik peremérték feladatok megoldására, például véges differenciák módszere, próbafüggvények módszere (globális maradék minimalizálása, lokális maradékok eliminálása). Ezek ismertetésére most idő hiányában nem térünk ki.

Természetesen a Matlab-nak van saját beépített parancsa is a peremérték feladatok megoldására, a **bvp4c** parancs (bvp = boundary value problem). Nézzünk most egy példát tartószerkezet témaköréből!

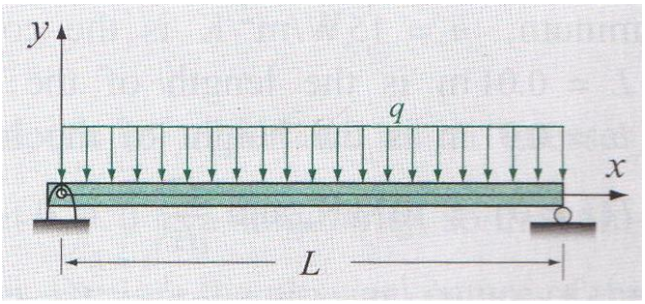
Az alábbi L=4 m hosszú kéttámaszú tartóra egyenletesen megoszló terhelés hat (q). Nagy lehajlásokra, a tartó lehajlása (y) a következő közönséges differenciál egyenlettel határozható meg:

$$EI\frac{d^2y}{dx^2} = \left[1+\left(\frac{dy}{dx}\right)^2\right]^{\frac{3}{2}} \cdot \frac{1}{2}\cdot q\cdot(L\cdot x - x^2),$$

$y(0)=0; y(L)=0;$ - peremfeltételek

$EI = 1.4\times 10^7 Nm^2$; - hajlítási merevség

$q = 10\times 10^3 N/m$; - terhelés



Oldjuk meg a feladatot a Matlab beépített függvényével!

1) Határozzuk meg a lehajlás értékeket *x* függvényében és jelenítsük meg őket! Jelenítsük meg az első deriváltakat is! Mennyi lesz a lehajlás értéke 1.35 m-nél?
2) Mennyi a maximális lehajlás értéke?
3) Milyen x értékeknél lesz a lehajlás pontosan 1 mm?

A Matlab beépített **bvp4c** peremérték feladat megoldó függvényét a következő formában lehet megadni:

```
> SOL = bvp4c(ODEFUN,BCFUN,SOLINIT)
```

A **bvp4c**-nek egy kimenete van (most ez a solution rövidítéseként a sol, de bármilyen változónév lehet), ami egy struktúra típusú változóként tartalmazza a független változót (sol.x) és a kiszámolt függő változó és derivált értékeket is (sol.y), és három bemenete:

- *odefun*: elsőrendű differenciálegyenlet rendszer (függvény)
- *bcfun*: a peremértékek függvényként megadva
- *solinit*: kiértékelendő tartomány, illetve a függvény és deriváltjai átlagos értékének becslése (a **bvpinit** paranccsal előállítva)

ELSŐRENDŰ DIFFERENCIÁLEGYENLET RENDSZER MEGADÁSA (ODEFUN)

Ehhez is, mint a kezdeti értékeknél használt **ode45** parancsnál át kell alakítani a magasabb rendű differenciálegyenletet egy elsőrendű differenciálegyenlet rendszerré (*odefun*). Ehhez először fejezzük ki a második deriváltat a másodrendű differenciálegyenletből!

$$\frac{d^2y}{dx^2} = \left(\left[1 + \left(\frac{dy}{dx} \right)^2 \right]^{\frac{3}{2}} \cdot \frac{1}{2} \cdot q \cdot (L \cdot x - x^2) \right) \frac{1}{EI}$$

Alakítsuk át egy új kételemű vektorváltozó bevezetésével elsőrendű differenciálegyenlet rendszerré! Legyen most: $w_1 = y; w_2 = \frac{dy}{dx}$.

$$w = \begin{pmatrix} y & \frac{dy}{dx} \end{pmatrix}$$

Ekkor az elsőrendű differenciálegyenlet rendszer:

$$f_1 = \frac{dw_1}{dx} = \frac{dy}{dx} = w_2$$

$$f_2 = \frac{dw_2}{dx} = \frac{d^2y}{dx^2} = \left([1 + w_2^2]^{\frac{3}{2}} \cdot \frac{1}{2} \cdot q \cdot (L \cdot x - x^2) \right) \frac{1}{EI}$$

Definiáljuk a differenciálegyenlet rendszert egy külön függvényben. Ezt megtehetjük egy külön fájlban, ahogy eddig is tettük, vagy a script fájl végén is (a Matlab R2016b verziójától). Legyen a függvény neve **diff_lehajlas**!

```
> function F = diff_lehajlas(x,w)
>   EI = 1.4e7; q = 10e3; L = 4;
>   f1 = w(2);
>   f2 = ((1+(w(2))^2)^(3/2) * (1/2) * q * (L*x-x^2) ) / EI;
>   F = [f1; f2];
> end
```

PEREMÉRTÉKEK FÜGGVÉNYKÉNT MEGADVA (BCFUN)

A peremfeltételek (a lehajlás az alátámasztásoknál):

$$y(0) = 0; \qquad y(L) = 0;$$

A **bvp4c** számára a peremfeltételeket is egy függvényben kell megadni (*bcfun*). Az [a,b] tartományon a feltételeket a wa és wb vektorokban adhatjuk meg, wa a tartomány elején, wb pedig a tartomány végén megadott feltételeket tartalmazza.

$$wa = \begin{pmatrix} y(a) & \left.\frac{dy}{dx}\right|_{x=a} & \cdots \end{pmatrix}; \; wb = \begin{pmatrix} y(b) & \left.\frac{dy}{dx}\right|_{x=b} & \cdots \end{pmatrix};$$

A vektorok első eleme a keresett függvény értéke a kezdő/végpontban (y), a második eleme az első derivált értéke $\left(\frac{dy}{dx}\right)$ a kezdő/végpontban és így tovább a 3., 4. stb. elem a második, harmadik stb. derivált értéke lenne. Jelen esetben mivel a keresett lehajlás (elmozdulás) függvény értékei adottak a kezdő és végpontban:

$$wa(1) = 0, \qquad wb(1) = 0$$

A peremértékek vektorfüggvényét nevezzük g-nek. Megadása úgy történik, mint zérushely kereséskor, amikor a 0-ra rendezett alakot kell megadni, tehát

$$g_1 = wa(1) - 0 = 0$$

$$g_2 = wb(1) - 0 = 0$$

A g függvény tulajdonképpen a maradék ellentmondásokat adja meg, ha ezek 0-k, akkor teljesíti a függvény a megadott feltételeket. Jelen esetben mivel pont 0 a peremérték mindkét helyen ezért lehetne egyszerűen $g_1 = wa(1),\ \ g_2 = wb(1)$.

Legyenek a peremfeltételek a **perem_lehajlas** függvényben (ez lehet külön fájlban vagy a script végén). Mivel most a lehajlás értékekre van feltételünk mindkét végpontban, ezek az *wa* és *wb* első elemei adottak.

```
function G=perem_lehajlas(wa,wb)
    g1=wa(1)-0;
    g2=wb(1)-0;
    G = [g1; g2];
end
```

Természetes matematikailag a '-0'-t nem kell beleírnunk a függvénybe, de így könnyebb megjegyezni, hogy máskor ide kell beírni a kezdeti értékeket, ha éppen eltérnek 0-tól. Ha mondjuk a kezdőpontban az első derivált értéke lenne megadva, például $\frac{dy}{dx}\Big|_{x=a} = 2$, a végpontban pedig a keresett függvény értéke $y(b) = 3$, akkor a megadás a következő lenne: $g_1 = wa(2) - 2,\ \ g_2 = wb(1) - 3$.

KIÉRTÉKELENDŐ TARTOMÁNY, ISMERETLENEK ÁTLAGOS ÉRTÉKE (SOLINIT)

A harmadik szükséges bemenet a **bvp4c** függvény használatához a kiértékelendő tartomány (*solinit*), illetve a függvény és deriváltjai átlagos értékének becslése (konstansként), amiket a **bvpinit** paranccsal kell megadni. A tartományunk most $0 \leq x \leq 4$. Vegyünk fel ezen 10 cm-ként pontokat. Meg kell még becsülni a függvény és az első deriváltjának is az átlagos értékét. Mivel most csak a függvény értékére van adatunk a kezdő és végpontban, és ezek mindegyike 0, legyen a becsült értékünk is 0 mind a függvényre, mind az első deriváltra. (Ha a kezdő és végpontban eltérő lenne az érték, akkor célszerű lenne a kettő átlagát választani itt a konstans becslésére.) A megoldást a **lehajlas_megoldas.m** fájlban készítsük el!

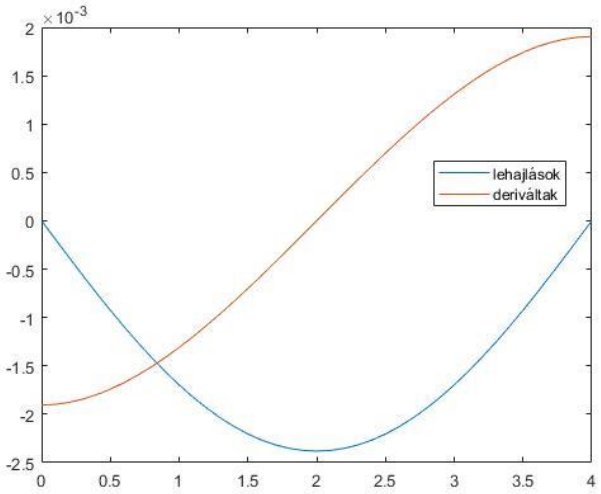
```
clc; clear all; close all;
% Becsült kezdeti értékek -  bvpinit
x0 = 0:0.1:4; % kiértékelendő tartomány
w10 = 0; w20 = 0; % becsült konstans értékek a függvényre és az első deriváltra
kezdofelt = bvpinit(x0,[w10; w20])
```

MEGOLDÁS BVP4C PARANCCSAL

A megoldáshoz adjunk meg 10^{-4} relatív toleranciát. Ezt nem az **odeset**, hanem a **bvpset** paranccsal tehetjük meg. Utána oldjuk meg a feladatot a **bvp4c** parancsot használva és ábrázoljuk is a megoldást!

```
% megoldás a beépített bvp4c függvénnyel
opciok = bvpset('RelTol',1e-4);
sol = bvp4c(@diff_lehajlas,@perem_lehajlas,kezdofelt,opciok)
X = sol.x;W = sol.y; % struktúra típusú megoldás elemei
Y = W(1,:); DY = W(2,:) % függvény és deriváltja
plot(X,Y); hold on; plot(X,DY);
legend('lehajlások','deriváltak', 'Location','best')
```

Mivel a megoldás struktúra típusú, így sol.x és sol.y hivatkozással tudjuk szétválasztani a független változót (X) tartalmazó vektort és a keresett függvényt és első deriváltjait tartalmazó (W) mátrixot.



Az eddigi megoldás egyben a következőképp néz ki:

```
> % Kétámaszú tartó lehajlása
> clear all; clc; close all;
> % Becsült kezdeti értékek -  bvpinit
> x0 = 0:0.1:4; % kiértékelendő tartomány
> w10 = 0; w20 = 0; % becsült konstans értékek a függvényre és az első
  deriváltra
> kezdofelt = bvpinit(x0,[w10; w20])
> % megoldás a beépített bvp4c függvénnyel
> opciok = bvpset('RelTol',1e-4);
> sol = bvp4c(@diff_lehajlas,@perem_lehajlas,kezdofelt,opciok)
> X = sol.x;
> Y = sol.y;
> plot(X,Y(1,:)); hold on
> plot(X,Y(2,:));
> legend('lehajlások','deriváltak', 'Location','best')
> %--------------------------------------
> function F = diff_lehajlas(x,w)
>   EI = 1.4e7; q = 10e3; L = 4;
>   f1 = y(2);
>   f2 = ((1+(w(2))^2)^(3/2) * (1/2) * q * (L*x-x^2) ) / EI;
>   F = [f1; f2];
> end
> %--------------------------------------
> function G=perem_lehajlas(wa,wb)
>     g1=wa(1);
>     g2=wb(1);
>     G = [g1; g2];
> end
> %--------------------------------------
```

Kérdés volt még, hogy mennyi lesz a lehajlás értéke 1.35 m-nél, mennyi a maximális lehajlás értéke és milyen x értékeknél lesz a lehajlás pontosan 1 mm?

Ezeket megoldhatnánk úgy is, hogy egy spline függvényt illesztünk a kapott pontokra, de használhatjuk a **deval** parancsot is, ami tetszőleges pontban kiértékeli a bvp4c

megoldását (**ode45** esetén is használható, ha azt nem két kimenettel hívjuk meg, hanem eggyel, akkor az is egy struktúra típusban adja vissza az értékeket). A **deval** csak interpolációra használható, extrapolációra nem!

```
> % Mennyi lesz a lehajlás értéke 1.35 m-nél?
> e1 = deval(sol,1.35) %-0.0021, -0.0009
> e1 = deval(sol,1.35,1) %-0.0021
> % Tehát 2.1 mm a lehajlás
> hold on; plot(1.35,e1,'ro')
```

Csak a megoldás és a kiértékelendő hely megadásakor a **deval** függvény használata során megkapjuk az adott pontban a függvény értékét, és az első derivált értékét is, ha csak az első (vagy a második) érdekel minket ezek közül, akkor megadhatjuk még a minket érdeklő változó indexét is. Most a lehajlás 2.1 mm lett a kérdéses pontban.

Mekkora lesz a maximális lehajlás? Ehhez definiáljuk függvényként a kapott megoldást! Mivel a koordináta rendszer úgy van felvéve, hogy a lehajlásokat negatív számként kapjuk, igazából egy minimumot keresünk, így egyszerűen használjuk most az **fminsearch** parancsot erre! Ehhez kezdőértéknek adjuk meg a 2-t!

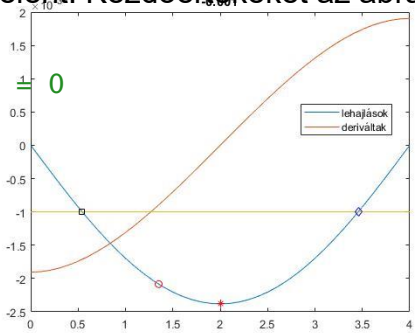
```
> % Mekkora lesz a maximális lehajlás?
> lehajlas = @(x) deval(sol,x,1)
> xmax = fminsearch(lehajlas,2) % 2
> lmax = lehajlas(xmax) % -0.0024
```

Mivel itt a terhelés teljesen szimmetrikus volt, egyértelmű volt, hogy a felezőpontban (2 méternél) lesz a maximális lehajlás, így akár le is kérdezhettük volna ott az értéket a **deval** használatával:

```
> lmax = deval(sol,2,1) %  -0.0024
> plot(xmax,lmax,'r*')
```

Milyen x értékeknél lesz a lehajlás pontosan 1 mm? Ez két pontot is jelenthet, melyeket az **fzero** használatával kereshetünk meg. Figyeljünk ismét a koordináta rendszerre és a mértékegységekre, hogy 1 mm lehajlás itt -0.001 m-t jelent! Kezdőértékeket az ábra alapján válasszunk 0.5 illetve 3.5 m-t!

```
> % lehajlas = -0.001 -> h = lehajlas + 0.001 = 0
> h = @(x) lehajlas(x) + 0.001
> x01 = 0.5; x02 = 3.5;
> x1 = fzero(h,x01) % 0.5437
> x2 = fzero(h,x02) % 3.4563
> plot(x1,lehajlas(x1),'ks'); hold on;
> plot(x2,lehajlas(x2),'bd')
```



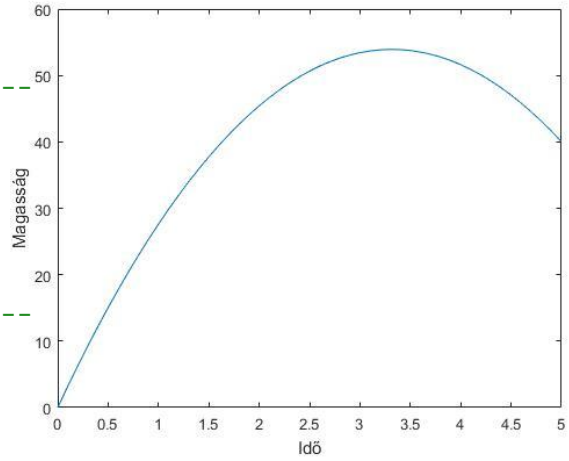
ELSŐ PÉLDA MEGOLDÁSA BVP4C FÜGGVÉNYT HASZNÁLVA

Nézzük meg hogyan nézne ki az elsőnek megoldott petárda kilövéses példánk a Matlab beépített függvényével! A másodrendű differenciálegyenlet: $\frac{d^2y}{dt^2} = -g$, a peremfeltételek: $y(0) = 0;\ y(5) = 40.$

A megoldás **bvp4c** használatával:

```
> % petárda - bvp4c
> clc; clear all; close all;
> % Becsült kezdeti értékek -  bvpinit
> t0 = 0:0.1:5; % kiértékelendő tartomány
```

```
w10 = 20; % becsült átlagos magasság (0+40)/2
w20 = 0; % becsült érték az átlagos első deriváltra
kezdofelt = bvpinit(t0,[w10; w20])
% megoldás a beépített bvp4c függvénnyel
opciok = bvpset('RelTol',1e-4);
sol = bvp4c(@diff_petarda,@perem_petarda,kezdofelt,opciok)
X = sol.x;
W = sol.y;
plot(X,W(1,:));
xlabel('Idő'); ylabel('Magasság')
%-------------------------------------
function F = diff_petarda(t,w)
  g = 9.81;
  f1 = w(2);
  f2 = -g;
  F = [f1; f2];
end
%-------------------------------------
function G=perem_petarda(wa,wb)
    g1=wa(1);
    g2=wb(1)-40;
    G = [g1; g2];
end
%-------------------------------------
```



GYAKORLÓ PÉLDA PEREMÉRTÉK FELADATOKHOZ

Oldjuk meg most a következő peremérték feladatot a [0,1] tartományon:

$$\frac{d^2y}{dx^2} + y = 0$$

Átrendezve:

$$\frac{d^2y}{dx^2} = -y$$

A megadott peremfeltételek: $y(0) = 1; \quad \frac{dy}{dx}(1) = 3$

Az intervallum elején adott a függvényérték, az intervallum végén pedig az első derivált! Az elsőrendű differenciálegyenlet rendszer, ha $w_1 = y; w_2 = \frac{dy}{dx}$:

$$f_1 = \frac{dw_1}{dx} = \frac{dy}{dx} = w_2$$

$$f_2 = \frac{dw_2}{dx} = \frac{d^2y}{dx^2} = -w_1$$

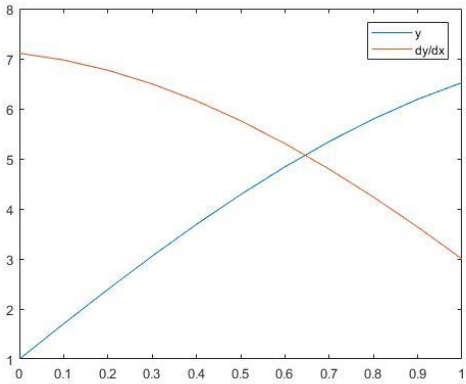
A peremfeltételek: $wa(1) = 1, \; wb(2) = 3$

Nullára rendezve: $g_1 = wa(1) - 1; \; g_2 = wb(2) - 3$

Megoldás Matlab-ban:

```
% gyakorló peremérték feladat
clc; clear all; close all;
clc; clear all; close all;
% Becsült kezdeti értékek -  bvpinit
```

```
x0 = 0:0.1:1; % kiértékelendő tartomány
w10 = 1; % becsült átlagos függvényérték
w20 = 2; % becsült érték az első deriváltra
kezdofelt = bvpinit(x0,[w10; w20])
% megoldás a beépített bvp4c függvénnyel
sol = bvp4c(@gyakorlo_diff,@gyakorlo_peremfelt,kezdofelt)
X = sol.x; W = sol.y;
plot(X,W(1,:),X,W(2,:));
legend('y','dy/dx')
%-----------------------------------
function F = gyakorlo_diff(x,w)
    f1 = w(2);
    f2 = -w(1);
    F = [f1; f2];
end
%-----------------------------------
function G=gyakorlo_peremfelt(wa,wb)
    g1=wa(1)-1;
    g2=wb(2)-3;
    G = [g1; g2];
end
%-----------------------------------
```



A FEJEZETBEN HASZNÁLT ÚJ FÜGGVÉNYEK

bvp4c	-	Közönséges differenciál egyenletek peremérték feladatának megoldása kollokációval
bvpinit	-	Kezdeti értékek becslése közönséges differenciálegyenletek peremérték feladatának megoldásához
bvpset	-	Közönséges differenciálegyenlet peremérték feladatát megoldó függvények opcionális paramétereinek megadása (pl. RelTol, AbsTol)
deval	-	Közönséges differenciálegyenlet megoldásának kiértékelése adott pontban