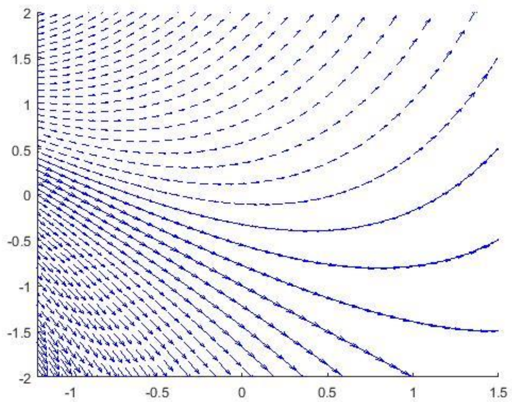
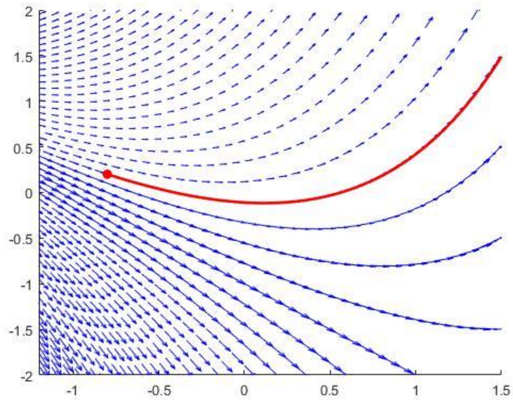
15. DIFFERENCIÁLEGYENLETEK – KEZDETI ÉRTÉK PROBLÉMA

A differenciálegyenlet egy olyan egyenlet, amely az ismeretlen függvény deriváltjait is tartalmazza. A megoldás itt nem egy konkrét érték lesz, hanem egy olyan függvény, amely kielégíti a differenciálegyenlet rendszert. Közönséges differenciálegyenlet esetén ez egy egyváltozós függvény.

Az egyértelmű megoldás érdekében meg kell adni egy (vagy több) pontot, amelyen a megoldásfüggvény áthalad. Nézzünk egy egyszerű példát, adott a következő differenciálegyenlet: $\frac{dy}{dx} = y + x$, keressük azt a függvényt, amelyik kielégíti ezt a feltételt. A lenti ábra bal oldalán ábrázolt összes függvény kielégíti ezt a feltételt. Ez a differenciálegyenlet trajektória vagy iránymezője (phase portrait). Ha azonban azt a megoldást keressük, ami áthalad az $x = -0.8, y = 0.2$ ponton, akkor már egyetlen függvény lesz a megoldásunk (jobb oldali ábrán pirossal jelölve).





Kezdeti érték probléma esetén ismerjük a függvény és deriváltja(i) értékét a kezdőpontban, ezek alapján próbáljuk meghatározni a függvényt. Peremérték feladatok esetében legalább az egyik érték (a függvény és deriváltjainak értékei közül) nem a kezdőpontban, hanem a végpontban adott. Ez azzal bonyolítja a feladatot, hogy meg kell határoznunk azt a kezdeti értéket is, ahonnan elindulva a végpontban megadott értéket kapjuk.

Lineáris differenciálegyenletben a keresett függvénynek vagy deriváltjának csak a lineáris kifejezése szerepel. Például:

$$e^x \frac{dy}{dx} + a \cdot x^2 + x^4 \cdot y = 0 - \text{lineáris differenciálegyenlet}$$

$$\frac{dy}{dx} + a \cdot x \cdot y + b \cdot y^2 = 0 - \text{nemlineáris differenciálegyenlet}$$

Az egyenlet n-ed rendű, ha abban az ismeretlen függvény legmagasabb deriváltja az n-edik derivált. A megoldásfüggvény meghatározása sokszor - különösen nemlineáris esetben - csak numerikusan lehetséges. Ebben az esetben a függvényt nem analitikusan kapjuk meg, hanem diszkrét pontokban a függvény értékeket, numerikus integrálással. A cél olyan numerikus eljárások alkalmazása, amelyek előírt lokális hiba mellett minél kevesebb lépéssel, pontosabb függvénykiértékeléssel képesek meghatározni a megoldásfüggvény pontjait.

ELSŐRENDŰ KÖZÖNSÉGES DIFFERENCIÁLEGYENLET-KEZDETIÉRTÉK PROBLÉMA

Elsőrendű differenciálegyenlet általános alakja (legyen *t* a független változó):

$$y' = \frac{dy}{dt} = f(t,y) \qquad \text{és} \qquad y(t_0) = y_0$$

Egyváltozós esetben egy független változónk van, ez most t, és egy függő változó, ezt most y -nal jelöltük. $f(t,y)$ függvény írja le az első deriváltat. Amennyiben a differenciálegyenlet bal oldalán nem csak az első derivált szerepel, akkor a megoldás előtt át kell rendezni az egyenletet a fenti alakra. Kezdeti érték probléma esetén kezdeti feltételként ismert, hogy a megoldás áthalad a (t_0, y_0) ponton:

EULER-MÓDSZER

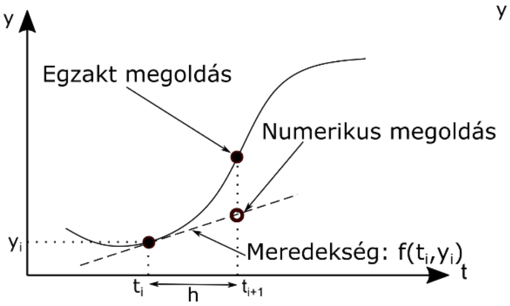
Szeretnénk meghatározni egy általunk felvett intervallumban, adott lépésközönként (h) az eredeti függvény értékeit. Tekintsük állandónak egy adott h szakaszon a függvény meredekségét (m). Ha ismerjük a függvény értékét a szakasz kezdőpontjában és a meredekség értékét, akkor a szakasz végén a függvény értékét közelíthetjük az ismert kezdőponton áthaladó m meredekségű egyenessel.

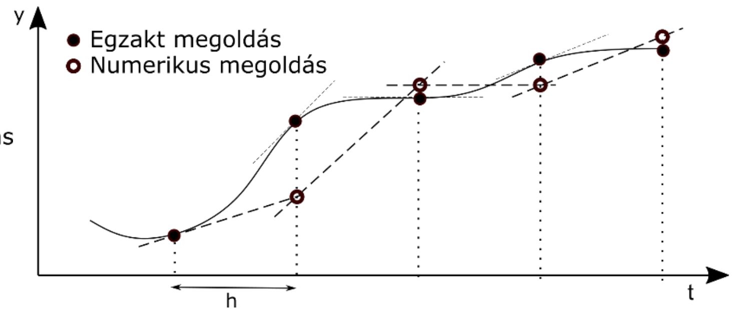
Az Euler-módszer esetén feltételezzük, hogy $m = f(t,y)$ értéke állandó az integrálási részintervallumokban ($h = t_{i+1} - t_i$) és értéke az intervallum elején kiszámolható értékkel egyezik meg.

$$y_{i+1} = y_i + \int_{t_i}^{t_{i+1}} f(t,y) \cdot dt \approx y_i + f(t_i, y_i) \cdot h = y_i + m_i \cdot h$$

$$t_{i+1} = t_i + h$$

ahol m a szakasz kezdőpontjában kiszámolt, az adott szakaszon állandónak tekintett meredekség. A módszer lokális hibája O(h^2), globális hibája pedig O(h), azaz a módszer elsőrendű.





Nézzük meg, hogyan oldhatjuk meg az Euler-módszert Matlab-ban (**euler.m**)!

```
> function [t,y] = euler (f, y0, a, b, h)
> n = round((b - a)/h);
> t(1) = a;
> y(1) = y0;
> for i = 1 : n
>     y(i + 1) = y(i) + h*f(t(i), y(i));
>     t(i + 1) = t(i) + h;
> end
```

A fenti függvény bemenő paraméterei:

- f – az elsőrendű differenciál egyenlet
- y0 – a megoldás függvény értéke a kezdőpontban
- a – az intervallum eleje
- b – az intervallum vége
- h – lépésköz nagysága a számítához

ELSŐRENDŰ DIFFERENCIÁLEGYENLET MEGOLDÁSA EULER-MÓDSZERREL

Nézzünk egy példát rá! Egy víztorony R=10 m sugarú gömb alakú tartályán alul, h=0 magasságban elhelyezkedő r=5 cm sugarú nyíláson keresztül elkezdik leengedni a benne tárolt (kb. 4000 m^3) vizet. A leengedés kezdetekor ($t = 0$) a vízszint magassága a tartályban 17.44 m. A nyílás kifolyási tényezője $\mu = 0.85$.

1) Mekkora lesz a vízszint a tartályban 12 óra múlva?
2) Mennyi idő alatt ürül ki a tartály?

A víztorony pillanatnyi vízszintjét (a tartály aljától mérve) a következő elsőrendű közönséges differenciálegyenlettel írhatjuk le:

$$f(t,h) = \frac{dh}{dt} = -\frac{\mu\, r^2 \sqrt{2\, g\, h}}{2\, h\, R - h^2}$$

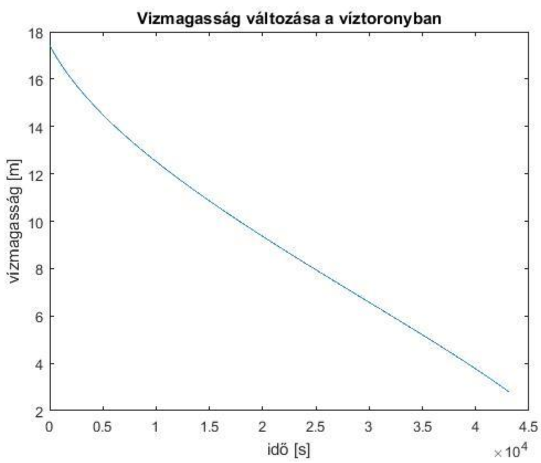
ahol $R = 10\ m, r = 0.05\ m, g = 9.81\,\frac{m}{s^2}, \mu = 0.85$.

1) Mekkora lesz a vízszint a tartályban 12 óra múlva?

Egy elsőrendű differenciálegyenletnél a megoldás első lépése mindig az, hogy kifejezzük az első deriváltat a többi változó függvényében, ha eredetileg nem így volt megadva. Ez lesz az *f* függvényünk.

Írjuk be a feladatot Matlab-ba és oldjuk meg Euler módszert használva, 60 másodperces lépésközzel! Előfordulhat, hogy az első derivált *f* függvényében nem szerepel a független változó (ami most *t*), de a megoldáshoz a Matlab-ban a differenciálegyenlet függvényének megadásakor az ismeretlenek között mindig meg kell adni a független változót is. Ez a helyzet most is, *t* csak a változók felsorolásánál szerepel, a függvényben nem.

```
>  R = 10; r = 0.05; g = 9.81; mu = 0.85;
>  % Derivált függvény megadása
>  f = @(t,h) -
   mu*r^2*sqrt(2*g*h)./(2*h*R-h.^2)
>  % Kezdeti feltétel, tartomány
>  h0 = 17.44; t0 = 0; tv = 12*3600 %
   12 óra = 43200 s
>  % lépésköz 60 s
>  d = 60;
>
>  % megoldás Euler-módszerrel
>  [T, H] = euler(f, h0, t0, tv, d);
>  figure(1)
>  plot(T,H);
>  xlabel('idő [s]')
>  ylabel('vízmagasság [m]')
```



```
> title('Vizmagasság változása a víztoronyban')
```

A H vektor utolsó eleme megadja, hogy mennyi volt a vízszint 12 óra elteltével:

```
> H(end) % 2.7712 m
```

Az Euler módszer elsőrendű, azaz O(h) hibájú módszer. Nézzük meg, tudunk-e ennél pontosabb módszert használni!

EULER MÓDSZER JAVÍTÁSAI (HEUN-, KÖZÉPPONTI-, RUNGE-KUTTA-MÓDSZER)

Hasonlóképp becsüli a függvény értékeket az Euler, a Heun, a Középponti és a Runge-Kutta módszer is, a különbség csupán a meredekség kiszámításának módjában van. Euler módszernél a lépésköz elején számoljuk ki a derivált értékét és ezt használjuk meredekségnek (lásd a korábbi ábrákat).

A **Heun módszernél** a meredekség az intervallum elején (m_i) és végén (m_{i+1}) számolt meredekségek átlaga. Ahhoz azonban, hogy a meredekséget az intervallum végén ki tudjuk számolni, ismerni kell az ottani függvény értéket is, mivel $m_{i+1} = f(t_{i+1}, y_{i+1})$. Ezért először egy ún. prediktor lépésként Euler módszerrel számítják a végpontbeli közelítő függvény értéket és ezt használják a meredekség meghatározásához. A két meredekség átlagát használva számítható a tényleges függvényérték a végpontban.

1) Prediktor lépés (Euler módszer): $y_{i+1}^{(0)} = y_i + m_i \cdot h = y_i + f(t_i, y_i) \cdot h$,
2) Korrektor lépés: $t_{i+1} = t_i + h$, $\quad m_{i+1} = f\left(t_{i+1}, y_{i+1}^{(0)}\right)$

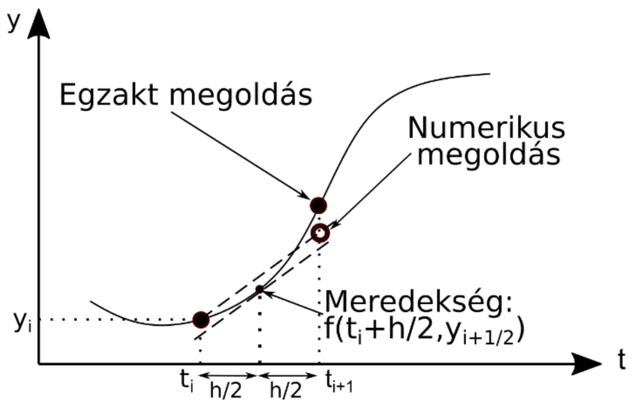
$$y_{i+1} = y_i + \frac{(m_i + m_{i+1})}{2} \cdot h = y_i + \frac{f(t_i, y_i) + f\left(t_{i+1}, y_{i+1}^{(0)}\right)}{2} \cdot h$$

A módszer lokális hibája O(h^3) és globális hibája O(h^2) azaz a módszer másodrendű hibájú, egy nagyságrenddel pontosabb, mint az Euler-módszer.

A **középponti módszer** esetén a felezőpontban számoljuk ki a deriváltat, és ez lesz az állandónak tekintett meredekség az egész intervallumra. Ehhez először ki kell számolni az előzetes függvényértéket a felezőpontban Euler módszerrel és utána tudjuk számolni ebben a pontban a meredekséget, amivel a végpontbeli függvényértéket kapjuk.

1) Felezőpont függvényértéke (Euler módszer): $y_{i+\frac{1}{2}} = y_i + m_i \cdot \frac{h}{2} = y_i + f(t_i, y_i) \cdot \frac{h}{2}$,
2) Meredekség a felezőpontban: $t_{i+\frac{1}{2}} = t_i + \frac{h}{2}$, $m_{i+\frac{1}{2}} = f\left(t_{i+\frac{1}{2}}, y_{i+\frac{1}{2}}\right)$
3) Függvényérték a végpontban: $y_{i+1} = y_i + m_{i+\frac{1}{2}} \cdot h$

A módszer lokális hibája O(h^3) és globális hibája O(h^2), azaz hasonlóan a Heun módszerhez, ez is egy nagyságrenddel pontosabb, mint az Euler-módszer.



Tovább lehet pontosítani az Euler-módszert, ha több pontban számoljuk ki a deriváltat, és ezek súlyozott átlaga lesz az állandónak tekintett meredekség. Ez a legelterjedtebb, negyedrendű hibájú **Runge-Kutta módszer**, amelynek globális csonkítási hibája: O(h^4). Matlab-ban ezt valósítja meg a beépített **ode45** függvény.

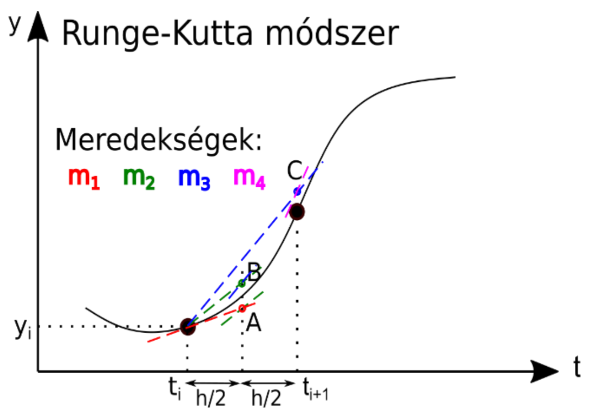
$$y_{i+1} = y_i + \frac{1}{6} \cdot (m_1 + 2m_2 + 2m_3 + m_4) \cdot h$$

$m_1 = f(t_i, y_i)$ - meredekség a kezdőpontban → A pont számítása ezzel

$m_2 = f\left(t_i + \frac{h}{2}, y_i + m_1 \cdot \frac{h}{2}\right)$ - meredekség az A pontban → B pont számítása ezzel

$m_3 = f\left(t_i + \frac{h}{2}, y_i + m_2 \cdot \frac{h}{2}\right)$ - meredekség a B pontban → C pont számítása ezzel

$m_4 = f(t_i + h, y_i + m_3 \cdot h)$ - meredekség a C pontban



$$m_1 = f(t_i, y_i)$$

$$y_A = y_i + m_1 \cdot \frac{h}{2} \;\rightarrow\; m_2 = f(t_i + \frac{h}{2}, y_A)$$

$$y_B = y_i + m_2 \cdot \frac{h}{2} \;\rightarrow\; m_3 = f(t_i + \frac{h}{2}, y_B)$$

$$y_C = y_i + m_3 \cdot h \;\rightarrow\; m_4 = f(t_i + h, y_C)$$

$$y_{i+1} = y_i + \frac{1}{6} \cdot (m_1 + 2m_2 + 2m_3 + m_4) \cdot h$$

ELSŐRENDŰ DIFFERENCIÁLEGYENLET MEGOLDÁSA RUNGE-KUTTA-MÓDSZERREL

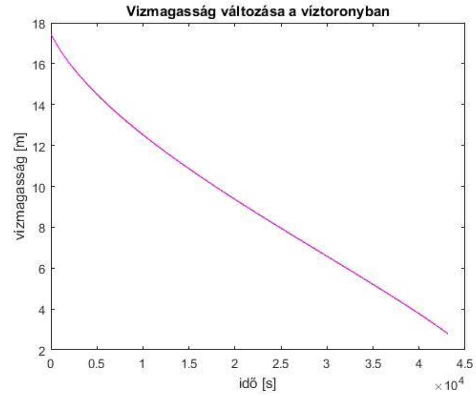
Oldjuk meg az előbbi víztornyos feladatot Runge-Kutta módszerrel is! Ehhez használjuk a Matlab beépített **ode45** parancsát!

Ennek legegyszerűbb hívása a következő:

```
>       [TOUT,YOUT] = ode45(ODEFUN,TSPAN,Y0)
```

ahol ODEFUN egy függvényhivatkozás $y' = f(t, y)$ függvényre. TSPAN lehet a [T_0 T_V] intervallum megadása a végpontokkal, vektor megadott lépésközökkel, illetve tetszőleges pontok egy vektorban. Y_0 az y függvény kezdeti értéke.

```
> % Megoldás Runge-Kutta-módszerrel
> [T1, H1] = ode45(f, [0,43200], h0);
> H1(end) % 2.7779 m
> % vagy lépésköz megadásával
> [T2, H2] = ode45(f, 0:60:43200, h0);
> H2(end) % 2.7713 m
> hold on;
> plot(T1,H1,'r')
> plot(T2,H2,'m')
```



Ebben az esetben nincs látható különbség a módszerek között, a végeredményben is csak pár mm az eltérés a vízszintben. Érdekes megnézni, hogyan vette fel az algoritmus a lépésközöket, abban az esetben, amikor csak kezdő és végső időpontot adtunk meg:

```
> diff(T1)
> min(diff(T1)) % 995.1333
> max(diff(T1)) % 1.1649e+03
```

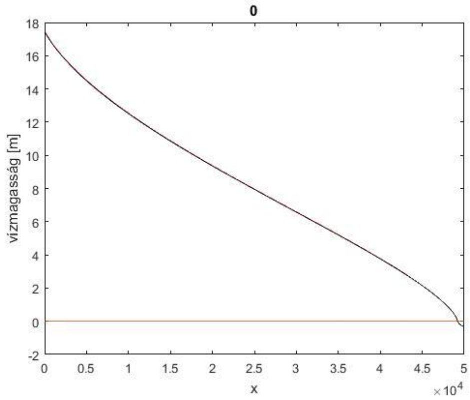
Tehát a lépésköz 995 és 1165 másodperc között változott. Sűrűbb lépésköz választása általában pontosítja az eredményt, viszont megnöveli a számítás időszükségletét.

Amennyiben tetszőleges időpontban szeretnénk megadni a víztartályban lévő aktuális vízszintet, akkor ezt megtehetjük a egyrészt úgy, hogy egy spline görbét illesztünk a pontokra, másrészt úgy is, hogy egy kimenettel hívjuk meg a függvényt, ami egy struktúra típus lesz. És ezzel a struktúra típusú megoldással használható a **deval** parancs, ami képes visszaadni a keresett függvény (és szükség esetén a deriváltjai) értékét a megadott pontban. Pl., ha arra vagyunk kíváncsiak, hogy 3 óra, vagyis 3*3600 másodperc múlva mekkora lesz a vízszint, azt a következőképpen is megtehetjük:

```
> % vízszint 3 óra (3*3600 másodperc) múlva
> sol = ode45(f,[t0,tv],h0)
> %   struct with fields:
> %      solver: 'ode45'
> %     extdata: [1×1 struct]
> %           x: [0 3980.5 8300.5 12621 16941 21261 25581 29901 34221
  38541 43200]
> %           y: [17.44 14.985 13.154 11.634 10.274 9.009 7.7984 6.6133
  5.4265 4.2044 2.7779]
> %       stats: [1×1 struct]
> %       idata: [1×1 struct]
> deval(sol,3*3600) %  12.249 m
```

2) Számítsuk ki, hogy a tartály teljes kiürüléséhez hány órára van szükség!

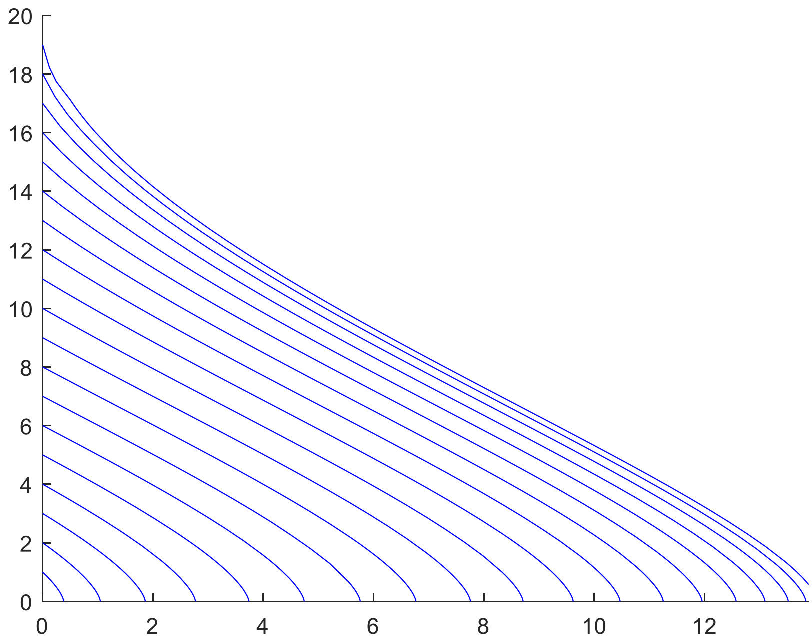
Azt láttuk, hogy 12 óra után még kb. 2.8 m volt a vízszint, tehát ennyi idő kevés volt a teljes kiürüléshez. Ezért először számítsuk ki hosszabb időre a lefolyás alakulását, mondjuk az előbbi 43200 másodperc helyett 50000 másodpercre. Vigyázzunk, mivel negatív *h* esetén komplex számokat kapunk megoldásként! A megoldáshoz illesszünk spline görbét a pontjainkra és zérushely kereséssel határozzuk meg a kiürülés időpontját!




```
> % Lépésköz megadásával
> [T3, H3] = ode45(f, 0:60:50000, h0)
> plot(T3,H3,'k')
> plot([0,50000],[0,0])
> % képzetes rész elhagyása
> H3 = real(H3);
> % spline illesztés
> sp =@(t) spline(T3,H3,t)
> fplot(sp,[0,50000])
> % Hány óra múlva lesz 0 a
  vízmagasság?
> x0 = fzero(sp,49000) % 4.9192e+04 s
> x0 = x0/3600 % 13.6644 h
```

Hogyan alakul a kiürülés különböző kezdeti magasságok esetén? Rajzoljuk fel a trajektória vagy iránymezőt, a maximális 20 méteres kiinduló vízszinttől egészen 1 méteres vízszintig. Az x tengelyt órában ábrázoljuk!

```
> % Trajektória vagy iránymező
> figure(2); hold on;
> for i=20:-1:1
>     [T, H] = ode45(f, [0,50000], i);
>     plot(T/3600,H,'b')
> end
> axis([0 50000/3600 0 20])
```



ELSŐRENDŰ DIFFERENCIÁLEGYENLET RENDSZER MEGOLDÁSA

Sok esetben egy adott folyamat több változótóval írható le, amelyek egymást is befolyásolhatják. Ilyen esetekben nem egy differenciálegyenletet, hanem egy differenciálegyenlet rendszert kell megoldanunk. Legyenek a függő változóink $y_1, y_2, \dots, y_n$, a független változónk pedig t. Általános esetben egy elsőrendű differenciálegyenlet rendszer felírása:

$$\frac{dy_1}{dt} = f_1(t, y_1, y_2, y_3 \dots, y_n)$$

$$\frac{dy_2}{dt} = f_2(t, y_1, y_2, y_3 \dots, y_n)$$

$$\dots$$

$$\frac{dy_n}{dt} = f_n(t, y_1, y_2, y_3 \dots, y_n)$$

A kezdeti értékek pedig az [a,b] tartományon:

$$y_1(a) = Y_1,\ y_2(a) = Y_2, \cdots, y_n(a) = Y_n,$$

Ezen egyenletrendszerek egy része megoldható a korábban ismertetett explicit módszerek általánosításával: $t_{i+1} = t_i + h$, Euler módszer esetében például:

$$y_{1,i+1} = y_i + f_1(t, y_1, y_2, y_3 \dots, y_n) \cdot h$$

$$\dots$$

$$y_{n,i+1} = y_i + f_n(t, y_1, y_2, y_3 \dots, y_n) \cdot h$$

Hasonlóképp általánosíthatóak az Euler módszer javításai és a Runge-Kutta módszer is. Nézzünk egy egyszerű kétváltozós példát erre.

A megoldást a [0, 1.2] tartományon keressük, *h*=0.4 lépésközönként.

$$\frac{dx}{dt} - x\,t + y = 0; \qquad x(0) = 1$$

$$\frac{dy}{dt} - y\,t - x = 0; \qquad y(0) = 0.5$$

Először rendezzük át az egyenleteket, hogy a baloldalon csak az első deriváltak szerepeljenek:

$$\frac{dx}{dt} = x\,t - y = f_1(t, x, y)$$

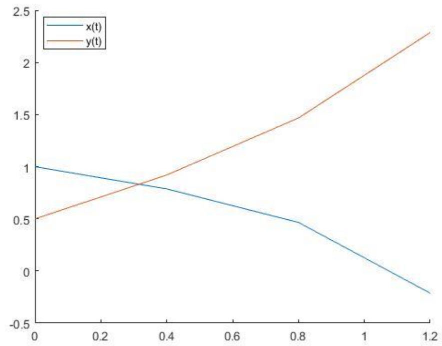
$$\frac{dy}{dt} = y\,t + x = \ f_2(t, x, y)$$

Itt két egyenletünk van, f_1 az egyik változó t szerinti első deriváltja, f_2 pedig a másik változó első deriváltja. Oldjuk meg a feladatot a Matlab beépített Runge-Kutta módszerével! A megadott x,y változók helyett vektorváltozót szükséges használni a Matlab beépített függvényeinek a hívásakor, legyen pl.

$v = [x; y]$, tehát $v_1 = x, v_2 = y$

Amennyiben nem túl bonyolult az egyenletrendszerünk, akkor megadhatjuk az egyenletrendszert egysoros függvényként a következőképp:

```
> f1 = @(t,v) v(1)*t-v(2)
> f2 = @(t,v) v(2)*t+v(1)
> F = @(t,v) [f1(t,v); f2(t,v)]
```



A megoldáshoz meg kell adni még a kezdőértékeket, értelmezési tartományt, lépésközt is.

```
> t = 0:0.4:1.2
> x0 = 1; y0 = 0.5; % kezdeti értékek
> [T,V] = ode45(F,t,[x0;y0])
> X = V(:,1); Y = V(:,2);
> figure(1); hold on; plot(T,X,T,Y)
> legend('x(t)','y(t)','Location','best')
```

Több változó vagy bonyolultabb összefüggések esetében már célszerű lehet külön fájlban megírni a differenciálegyenlet rendszert. Nézzük meg így is a megoldást. Írjuk meg egy külön **diffrsz.m** fájlba az elsőrendű differenciálegyenlet rendszert!

```
> function F = diffrsz(t,v)
>     f1 = v(1)*t - v(2);
>     f2 = v(2)*t + v(1);
>     F = [f1; f2];
> end
```

Figyeljünk oda, ha külön *.m fájlban adtuk meg a differenciálegyenlet rendszert, akkor a meghívásakor a függvény neve elé kell írni egy @ jelet!

```
> [T, V] = ode45(@diffrsz, t, [x0; y0])
```

MÁSODRENDŰ DIFFERENCIÁLEGYENLETEK

Egy másodrendű közönséges differenciálegyenlet t független és y függő változóval a következő alakba írható:

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

Az egyenlet megoldható [a,b] intervallumon, ha van két ismert feltételünk. Amennyiben a két megadott érték a tartomány elején van, akkor kezdeti érték feladatról beszélünk. A két kezdeti feltétel az y és $\frac{dy}{dt}$ értéke a kezdőpontban. Jelölje ezeket az értékeket A és B.

$$y(a) = A; \qquad \left.\frac{dy}{dt}\right|_{t=a} = B$$

Ez a fajta másodrendű differenciálegyenlet átalakítható két elsőrendű differenciálegyenletből álló egyenletrendszerré, ami az előzőekhez hasonlóan megoldható. A feladat megoldásához az első lépés, hogy kifejezzük a második deriváltat, amennyiben nem ilyen formában van megadva az egyenlet. A második deriváltat $f\left(t, y, \frac{dy}{dt}\right)$ függvényeként írjuk fel. Természetesen nem biztos, hogy ezek mindegyikétől függ. Itt t a független változó, ezt Matlab-ban akkor is meg kell adni, ha esetleg nem függ tőle közvetlenül a derivált függvény. A függő változó és deriváltjai helyett vezessünk be egy új vektorváltozót (w)!

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

Használjuk y *és* $\frac{dy}{dt}$ helyett w elemeit új változókként: $w_1 = y$ és $w_2 = \frac{dy}{dt}$. Ekkor két egyenletet kell felírnunk a két új változó első deriváltjaira, és ezekhez kell megadni a kezdőértékeket:

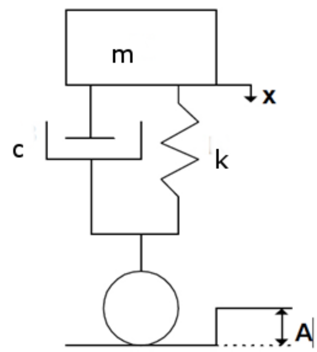
$$f_1 = \frac{dw_1}{dt} = \frac{dy}{dt} = w_2; \qquad w_1(a) = A$$

$$f_2 = \frac{dw_2}{dt} = \frac{d^2y}{dt^2} = f(t, w_1, w_2); \qquad w_2(a) = B$$

Ezekkel a definíciókkal a másodrendű differenciálegyenlet felírható két elsőrendű differenciálegyenletből álló egyenletrendszerként! Oldjunk meg egy ilyen másodrendű differenciálegyenletet!

MÁSODRENDŰ DIFFERENCIÁLEGYENLET MEGOLDÁSA MATLAB-BAN

Egy autó rugózásának szimulációját végezzük az alábbi egyszerű modell alapján, ahol az autó éppen áthalad egy *A* magasságú akadályon. A modellben *m* az autó tömege, *k* a rugómerevség (a rugóban fellépő erő arányos az elmozdulással), *c* a csillapítási tényező (a csillapító erő arányos a tömeg sebességével). Az adatok: $m = 1000\,kg; k = 1000\frac{kg}{s^2}; c = 500\frac{kg}{s};\ A = 0.1\,m$. A kiinduló időpontban mind az autó függőleges helyzete, mind a függőlege sebessége 0. A vizsgált időintervallum 15 másodperc.



Csillapított szabad rezgésnél a tömegre ható erőket összegezve az alábbi közönséges differenciálegyenletet kapjuk az autó függőleges mozgására:

$$m\,\ddot{x} + c\,\dot{x} + k\,x = 0$$

Ahol x az autó magassági helyzete, $\dot{x}$ az idő szerinti első derivált, tehát az autó függőleges sebessége, $\ddot{x}$ pedig az idő szerinti második derivált, vagyis az autó függőleges gyorsulása. Áttérve az autó koordináta rendszerére a függőleges irányú mozgás mozgásegyenlete:

$$m\,\frac{d^2x}{dt^2} + c\,\frac{dx}{dt} + k\,(x - A) = 0$$

A kezdeti feltételek, hogy a kezdeti függőleges helyzet és a kezdeti függőleges sebesség is nulla, mielőtt az akadályhoz érne az autó:

$$x(0) = 0; \qquad \left.\frac{dx}{dt}\right|_{x=0} = 0$$

Első lépésként fejezzük ki a második deriváltat $\left(\frac{d^2x}{dt^2}\right)$-t az egyenletből!

$$\frac{d^2x}{dt^2} = \frac{1}{m}\left(k\,A - k\,x - \,c\,\frac{dx}{dt}\right) = f\left(t, x, \frac{dx}{dt}\right)$$

Alakítsuk át a másodrendű differenciálegyenletet elsőrendű differenciálegyenlet rendszerré! Vezessünk be egy új vektor változót a függő változó és deriváltjai helyett:

$$w = \begin{pmatrix} x & \frac{dx}{dt} \end{pmatrix}$$

Használjuk a $w_1 = x$ és $w_2 = \frac{dx}{dt}$ új változókat az egyenletünkben! Két egyenletet kell felírnunk, a két új változó első deriváltjaira, és ezekhez kell megadni a kezdőértékeket:

$$f_1 = \frac{dw_1}{dt} = \frac{dx}{dt} = w_2; \qquad w_1(0) = 0$$

$$f_2 = \frac{dw_2}{dt} = \frac{d^2x}{dt^2} = \frac{1}{m}(k\,A - k\,w_1 - \,c\,w_2); \qquad w_2(0) = 0$$

Írjuk meg a differenciálegyenlet rendszert egy külön **autodiff.m** fájlban Matlab-ban! Legyen w egy vektorváltozó: $w = [w_1, w_2]$, tehát $w(1) = x$ a függőleges pozíció és $w(2) = \frac{dx}{dt}$ pedig a függőleges sebesség.

```
> function f = autodiff(t,w)
>    %  A mozgásegyenlet konstansai
>    m=1000; k=1000; A=0.1; c=500;
>    f1 = w(2);
>    f2 = 1/m*(k*A - k*w(1) - c*w(2));
>    f = [f1; f2];
> end
```

Figyeljük meg, hogy a bemenő változók között szerepel a *t* változó is, még akkor is, ha f_1,f_2 kifejezésben közvetlenül nem! Oldjuk meg a feladatot a Matlab beépített, Runge-Kutta módszert használó, **ode45** parancsával, 10^{-4} abszolút és relatív pontossággal, 0-15 másodpercre!

Az **ode45** opcionális paramétereit eddig még nem alkalmaztuk, de lehetőségünk van több érték beállítására az **odeset()** függvényt használva. A fontosabbak:

- RelTol = skalár relatív hibakorlát, amelyik az y minden komponensére érvényes
- AbsTol= skalár vagy vektor abszolút hibakorlát, amelyik a megoldásfüggvényekre egységesen vagy külön-külön érvényes
- MaxStep = maximális megengedett lépésköz
- InitialStep = javasolt kezdő *t* lépésköz

A megoldást készítsük el a **rezgomozgas.m** fájlba (fontos, hogy a megoldást tartalmazó fájl és a differenciálegyenlet rendszert tartalmazó fájl ugyanabban a könyvtárban legyen!):

```
> % Csillapított rezgés
> clc; clear all; close all;
> %  Megoldás Runge-Kutta módszerrel (ode45, odeset)
> options = odeset('RelTol', 1e-4, 'AbsTol',[1e-4 1e-4]);
> % legyen az időintervallum [0, 15] másodperc
> x0=0; % kezdeti pozíció
> v0=0; % kezdeti függőleges sebesség
> [T,W]=ode45(@autodiff,[0,15],[x0; v0],options);
```

A megoldásként kapott W mátrix első oszlopában vannak az elmozdulás értékek ($w(1) = x$) és a második oszlopában az első deriváltak ($w(2) = \frac{dx}{dt}$), vagyis a sebesség értékek.

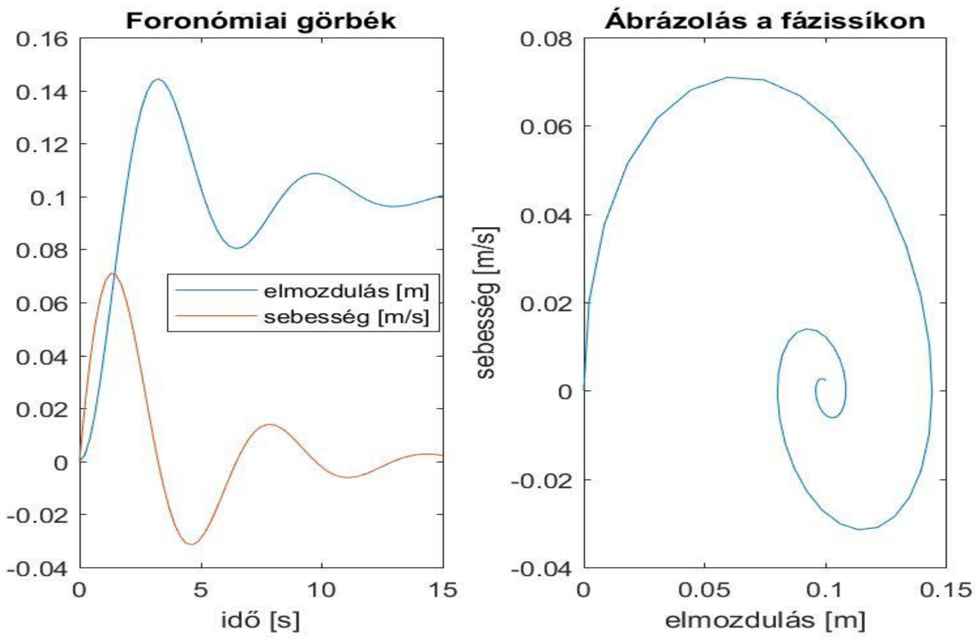
Mivel nem túl bonyolult egyenletrendszerről van szó a feladat megoldható lett volna egysoros függvény használatával is a következőképp:

```
> % Más megoldás egysoros függvény használatával
> m=1000; k=1000; A=0.1; c=500;
> dwdt = @(t,w) [w(2); 1/m*(k*A - k*w(1) - c*w(2))]
> options = odeset('RelTol', 1e-4, 'AbsTol',[1e-4 1e-4]);
> x0=0; v0=0;
> [T1,W1]=ode45(dwdt,[0,15],[x0; v0],options);
```

Megjegyzések: Mindkét megoldás egyenértékű, külön fájlban megírva a differenciálegyenlet rendszert szemléletesebb. Figyeljünk arra, hogy a differenciálegyenlet rendszerben nem szerepel külön a *t* paraméter, mégis meg kell adni a bemenő változóknál a differenciálegyenlet megoldásához! Ugyancsak fontos, ha a differenciálegyenlet rendszert külön fájlban adtuk meg, kell a neve elé írnunk egy @ jelet, ha egysoros függvényként, akkor nem.

Az elmozdulás, sebesség, gyorsulás értékek idő függvényében történő ábrázolását foronómiai görbéknek nevezik, a sebességek ábrázolását az elmozdulás függvényében (ahol az idő a görbe paramétere lesz) pedig fázis síkon történő ábrázolásnak. Rajzoljuk fel két egymás melletti ábrába a foronómiai görbéket és a fázissíkon a sebességeket az elmozdulás függvényében!

```
> % A kocsiszekrény elmozdulása és sebessége az idő függvényében
> subplot(1,2,1)
> x = W(:,1); % függőleges elmozdulás
> v = W(:,2); % függőleges sebesség
> plot(T,x,T,v)
> legend('elmozdulás [m]', 'sebesség [m/s]','Location','Best')
> xlabel('idő [s]')
> title('Foronómiai görbék')
>
> %  Ábrázolás a fázissíkon - az idő most paraméter
> % sebesség az elmozdulás függvényében
> subplot(1,2,2)
> plot(x,v)
> xlabel('elmozdulás [m]')
> ylabel('sebesség [m/s]')
> title('Ábrázolás a fázissíkon')
```



MÁSODRENDŰ DIFFERENCIÁLEGYENLET SZIMBOLIKUS MEGOLDÁSA MATLAB-BAN[1]

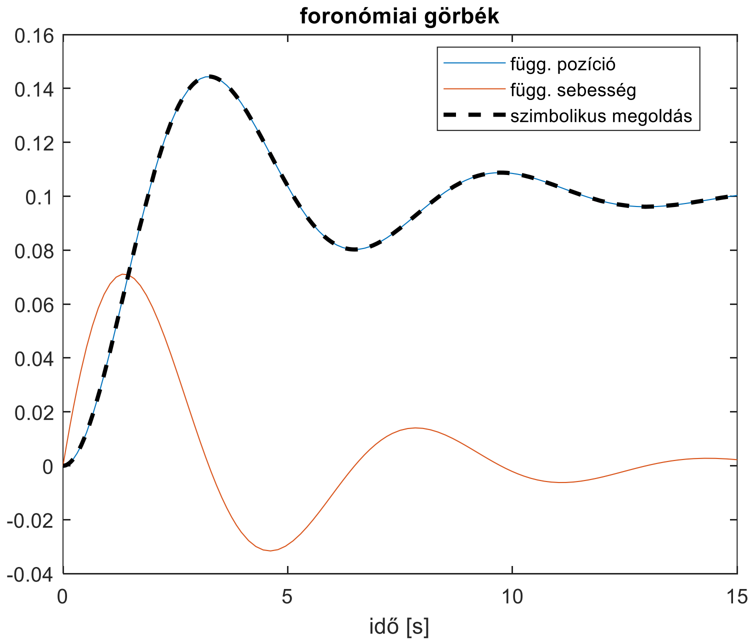
Bizonyos egyszerűbb differenciálegyenlet esetében lehetőség van szimbolikus megoldásra is a Matlab-ban, a 'Symbolic toolbox' **dsolve** parancsát használva. Sajnos azonban ez gyakran nem vezet eredményre, ilyenkor csak a numerikus megoldás jöhet szóba (pl. a korábban ismertetett víztartály differenciál egyenlete esetében is). Ebben az utoljára ismertetett másdofokú differenciálegyenlet esetében ($\frac{d^2x}{dt^2} + c\,\frac{dx}{dt} + k\,(x - A) = 0$) azonban működik a szimbolikus megoldás is. Az általános megoldást lásd a következőkben!

```
> % szimbolikus megoldás általánosan
> m = 1000; k = 1000; A = 0.1; c = 500;
> syms x(t)
> Dx = diff(x)
> % m*x"+ c*x' + k*(x-A) = 0
> ode = m*diff(x,t,2) + c*diff(x,t) + k*(x-A) == 0
> xSol(t) = dsolve(ode)
> % C1*exp(-t/4)*cos((15^(1/2)*t)/4) - C2*exp(-t/4)*sin((15^(1/2)*t)/4)
  + 1/10
```

Kezdeti feltételek megadásával a konkrét megoldás pedig így kapható meg:

```
> cond1 = x(0) == 0;
> cond2 = Dx(0) == 0;
> conds = [cond1 cond2];
> xSol(t) = dsolve(ode,conds)
> % 1/10 - (15^(1/2)*exp(-t/4)*sin((15^(1/2)*t)/4))/150 - (exp(-
  t/4)*cos((15^(1/2)*t)/4))/10
> xf = matlabFunction(xSol(t))
> figure(1); hold on; fplot(xf,[0,15],'k--','LineWidth',2)
> legend('függ. pozíció','függ. sebesség', 'szimbolikus megoldás',
  'Location','best')
```

[1] Otthoni átnézésre



MAGASABB RENDŰ DIFFERENCIÁLEGYENLETEK

Harmad-, negyed- vagy magasabb rendű differenciálegyenleteket szintén vissza lehet vezetni elsőrendű differenciálegyenlet rendszerre, hasonlóan a másodrendű esethez, új változók bevezetésével. Természetesen annyi kezdőértékre lesz szükségünk, ahány egyenletet felírunk, harmadrendű esetben 3, negyedrendű esetben 4 stb.

Egy *n*-ed rendű differenciálegyenlet általánosan:

$$\frac{d^n y}{dt^n} = f\left(t, y, \frac{dy}{dt}, \frac{d^2 y}{dt^2}, \dots, \frac{d^{(n-1)} y}{dt^{(n-1)}}\right), \qquad a \le t \le b$$

Kezdeti feltételek:

$$y(a) = A_1;\ \left.\frac{dy}{dt}\right|_{t=a} = A_2;\ \left.\frac{d^2 y}{dt^2}\right|_{t=a} = A_3; \dots ; \left.\frac{d^{(n-1)} y}{dt^{(n-1)}}\right|_{t=a} = A_n;$$

Vezessünk be egy n elemű új vektorváltozót:

$$w = \begin{pmatrix} y & \frac{dy}{dt} & \frac{d^2 y}{dt^2} & \dots & \frac{d^{n-1} y}{dt^{n-1}} \end{pmatrix}$$

Írjuk fel a következő elsőrendű differenciálegyenlet rendszert *w* elemeire a hozzájuk tartozó kezdeti feltételekkel együtt ($y = w_1$):

$$f_1 = \frac{dw_1}{dt} = \frac{dy}{dt} = w_2; \qquad w_1(a) = A_1$$

$$f_2 = \frac{dw_2}{dt} = \frac{d^2y}{dt^2} = w_3; \qquad w_2(a) = A_2$$

$$\ldots \qquad \ldots$$

$$f_{n-1} = \frac{dw_{n-1}}{dt} = \frac{d^{n-1}y}{dt^{n-1}} = w_n; \qquad w_{\mathrm{n}-1}(a) = A_{\mathrm{n}-1}$$

$$f_n = \frac{dw_n}{dt} = \frac{d^ny}{dt^n} = f\left(t, y, \frac{dy}{dt}, \frac{d^2y}{dt^2}, \ldots, \frac{d^{(n-1)}y}{dt^{(n-1)}}\right); \qquad w_n(a) = A_n$$

Példaként oldjuk meg a következő harmadrendű differenciálegyenletet a [0,1] intervallumon!

$$2x - 3y + 4\,\frac{dy}{dx} + x\,\frac{d^2y}{dx^2} - \frac{d^3y}{dx^3} = 0$$

Ahol a következő kezdeti feltételek adottak:

$$y(0) = 3;\ \left.\frac{dy}{dx}\right|_{x=0} = 2;\ \left.\frac{d^2y}{dx^2}\right|_{x=0} = 7;$$

Első lépés, hogy fejezzük ki a legmagasabb deriváltat!

$$\frac{d^3y}{dx^3} = 2x - 3y + 4\,\frac{dy}{dx} + x\,\frac{d^2y}{dx^2}$$

Alakítsuk át a harmadrendű differenciálegyenletet egy elsőrendű differenciálegyenlet rendszerré, ami 3 egyenletet tartalmaz! A harmadik deriváltat felírhatjuk $f\left(x, y, \frac{dy}{dx}, \frac{d^2y}{dx^2}\right)$ függvényeként. A függő változó és deriváltjai helyett vezessünk be egy új vektorváltozót!

$$w = \begin{pmatrix} y & \frac{dy}{dt} & \frac{d^2y}{dt^2} \end{pmatrix}$$

Tehát: $w_1 = y, w_2 = \frac{dy}{dx}, w_3 = \frac{d^2y}{dx^2}$. Az elsőrendű differenciálegyenlet rendszerben az újonnan bevezetett változók első deriváltjait kell megadjuk! 3 változónk van, tehát 3 egyenletet kell felírnunk.

$$f_1 = \frac{dw_1}{dx} = \frac{dy}{dx} = w_2; \qquad w_1(0) = 3$$

$$f_2 = \frac{dw_2}{dx} = \frac{d^2y}{dx^2} = w_3; \qquad w_2(0) = 2$$

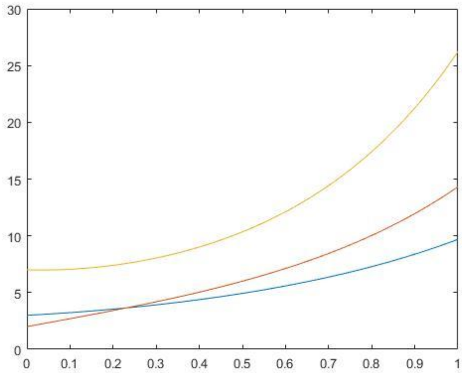
$$f_3 = \frac{dw_3}{dx} = \frac{d^2y}{dx^2} = 2x - 3w_1 + 4\,w_2 + x\,w_3; \qquad w_3(0) = 7$$

Matlab-ban ennek a felírása a **diff3.m** fájlban, *w=[w1,w2,w3]:*

```
> function dwdx = diff3(x,w)
>    f1 = w(2);
>    f2 = w(3);
>    f3 = 2*x - 3*w(1) +4*w(2) + x*w(3);
>    dwdx = [f1; f2; f3];
> end
```

Megoldása a [0,1] intervallumon:

```
> w10=3; w20=2; w30=7;
> [X,W]=ode45(@diff3,[0,1],[w10; w20; w30])
> figure(1);
> plot(X,W(:,1),X,W(:,2),X,W(:,3))
```



MAGASABB RENDŰ DIFFERENCIÁLEGYENLET RENDSZEREK

Egy magasabb rendű differenciálegyenlet rendszer hasonlóképp felírható új változók bevezetésével elsőrendű differenciálegyenlet rendszerré. Nézzünk például egy másodrendű differenciálegyenlet rendszert!

$$\frac{d^2x}{dt^2} = F_1\left(t, x, y, \frac{dx}{dt}, \frac{dy}{dt}\right)$$

$$\frac{d^2y}{dt^2} = F_2\left(t, x, y, \frac{dx}{dt}, \frac{dy}{dt}\right)$$

Definiáljunk egy új vektorváltozót a függő változók és deriváltjaik helyett!

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

Tehát: $w_1 = x$; $w_2 = y$; $w_3 = \frac{dx}{dt}$; $w_4 = \frac{dy}{dt}$; A négy új változónak megfelelő 4 egyenletből álló lineáris egyenletrendszer a következő lesz:

$$f_1 = \frac{dw_1}{dt} = \frac{dx}{dt} = w_3$$

$$f_2 = \frac{dw_2}{dt} = \frac{dy}{dt} = w_4$$

$$f_3 = \frac{dw_3}{dt} = \frac{d^2x}{dt^2} = F_1\left(t, x, y, \frac{dx}{dt}, \frac{dy}{dt}\right)$$

$$f_4 = \frac{dw_4}{dt} = \frac{d^2y}{dt^2} = F_2\left(t, x, y, \frac{dx}{dt}, \frac{dy}{dt}\right)$$

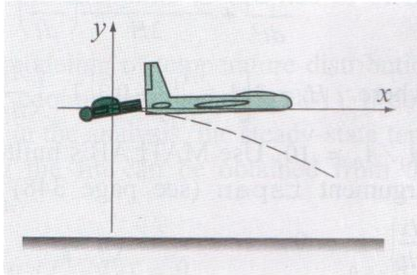
A megoldása az előzőek szerint történhet!

GYAKORLÓ FELADAT

Egy ejtőernyős kiugrik egy egyenesen, vízszintesen haladó repülőből. Az ejtőernyős mozgása közelítőleg az alábbi egyenletrendszerrel írható le:

$$\frac{d^2x}{dt^2} = -\frac{\gamma}{m}\left(\frac{dx}{dt}\right)\sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2}$$

$$\frac{d^2y}{dt^2} = -g - \frac{\gamma}{m}\left(\frac{dy}{dt}\right)\sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2}$$



ahol x és y az ejtőernyős pozíciója, az ábrán látható koordináta rendszernek megfelelően.

$m = 80\ kg; g = 9.81 m/s^2;\ \gamma\ = 5.38\ Ns^2/m^2$

A kezdeti feltételek:

$$x(0) = 0; y(0) = 0; \left.\frac{dx}{dt}\right|_{t=0} = 134\ m/s; \left.\frac{dy}{dt}\right|_{t=0} = 0;$$

Határozza meg az ejtőernyős mozgásának pályáját az első 5 másodpercben. A két másodfokú közönséges differenciál egyenletet vezesse vissza 4 elsőfokú közönséges differenciál egyenletre és oldja meg a feladatot tetszőleges módszerrel.

a) Vezesse vissza a feladatot új változók bevezetésével 4 elsőrendű differenciál egyenletre, és írja meg a rendszert reprezentáló Matlab függvényt! (Ez lehet egysoros függvény is, vagy külön függvényben is megadva)
b) Oldja meg az egyenletrendszert tetszőleges módszerrel az első 5 másodpercre, felhasználva a 4 kezdeti értéket!
c) Rajzolja fel az ejtőernyős pályáját az xy koordináta rendszerben! Két másik ábrába rajzolja fel a következő foronómiai görbéket: x,y elmozdulás és v_x, v_y sebességek az idő függvényében!
d) Mennyi 5 másodperc után az elmozdulás és a sebesség a két koordináta tengely irányában?
e) Mennyi a függőleges elmozdulás 2.25 másodperc után?
f) Az ejtőernyős 1500 m-es magasságban nyitja ki az ejtőernyőjét. Mikor éri el ezt a magasságot, ha a repülőgép az ugráskor 2000 m-en haladt? (Ehhez modellezze hosszabb intervallumra, pl. egy percre az ejtőernyős pályáját!)

<u>Megoldás</u>

a) Írjuk fel az elsőrendű differenciálegyenlet rendszert. Ehhez vezessünk be egy vektorváltozót (w) az ismeretlen x,y pozíciókra és az első deriváltjaikra. Ezek első deriváltjai adják az elsőrendű differenciálegyenlet rendszert.

$$w = \left(x, y, \frac{dx}{dt}, \frac{dy}{dt}\right)$$

$$f_1 = \frac{dw_1}{dt} = \frac{dx}{dt} = w_3$$

$$f_2 = \frac{dw_2}{dt} = \frac{dy}{dt} = w_4$$

$$f_3 = \frac{dw_3}{dt} = \frac{d^2x}{dt^2} = -\frac{\gamma}{m}\left(\frac{dx}{dt}\right)\sqrt{\left(\frac{\mathrm{dx}}{\mathrm{dt}}\right)^2 + \left(\frac{dy}{dt}\right)^2} = -\frac{\gamma}{m}w_3\sqrt{{w_3}^2 + {w_4}^2}$$

$$f_4 = \frac{dw_4}{dt} = \frac{d^2y}{dt^2} = -g - \frac{\gamma}{m}\left(\frac{dy}{dt}\right)\sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2} = g - \frac{\gamma}{m}w_4\sqrt{{w_3}^2 + {w_4}^2}$$

Írjuk fel Matlab-ban az egyenletrendszert, most egysoros függvényként:

```
> %% diff1 - ejtőernyős mozgása
> clc; clear all; close all;
> % elsőrendű diff. egyenlet rendszer anonymous függvényként
> g = 9.81; gamma = 5.38; m = 80;
> ernyodiff = @(t,w) [w(3);
>                  w(4);
>                 -(gamma/m)*w(3)*sqrt(w(3)^2+w(4)^2);
>                 -g-(gamma/m)*w(4)*sqrt(w(3)^2+w(4)^2)
```

b) Oldjuk meg a feladatot!

```
> % kezdeti értékek
>  x0 = 0; y0 = 0; % kezdeti x,y pozíciók
>  vx0 = 134; vy0 = 0; % kezdeti x,y sebességek
>
>  % Egyenletrendszer megoldása beépített függvénnyel az első 5
  másodpercben
> [T,W]=ode45(ernyodiff,[0,5],[x0; y0; vx0; vy0])
> X = W(:,1); Y = W(:,2); % pozíciók
> VX = W(:,3); VY = W(:,4); % sebességek
```

c) Rajzoljuk fel a megoldásokat!

```
> % Az ejtőernyős pályája xy koordináta rendszerben
> figure(1); plot(X,Y)
> title('Az ejtőernyős pályája az első 5 másodpercben')
> % x,y koordináta az első 5 másodpercben
> figure(2); plot(T,X,T,Y);
> legend('x koordináta','y koordináta','Location','best')
> title('Az x,y irányú elmozdulás az idő függvényében')
> % x,y irányú sebesség az első 5 másodpercben (foronómiai görbék)
> figure(3); plot(T,VX,T,VY);
> legend('x irányú sebesség','y irányú sebesség','Location','best')
> title('Az x,y irányú sebesség az idő függvényében')
```

d) Elmozdulás és sebesség 5 másodperc után

```
> % Elmozdulás és sebsesség 5 másodperc után a koordináta tengelyek
  irányában
> disp('vízszintes elmozdulás 5s után: '); disp(X(end)) % 50.1310
> disp('függőleges elmozdulás 5s után: '); disp(Y(end)) % -44.2489
> disp('vízszintes sebesség 5s után: '); disp(VX(end)) % 0.5648
> disp('függőleges sebesség 5s után: '); disp(VY(end)) % -12.0214
```

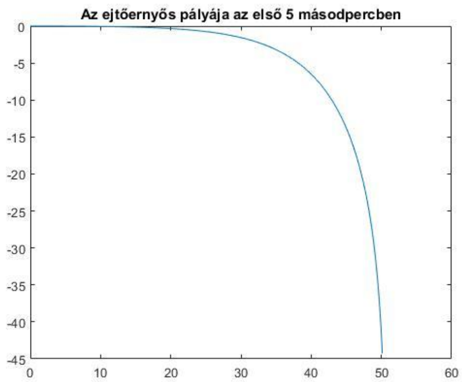
e) Mennyi a függőleges elmozdulás 2.25 másodperc után?

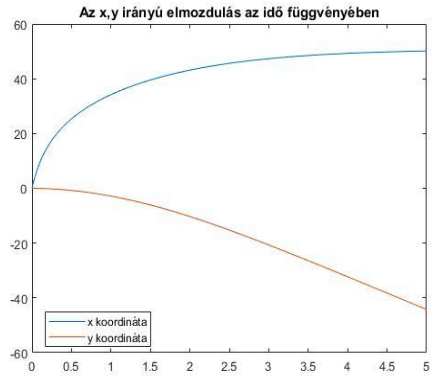
```
> % Mennyi függőleges elmozdulás 2.25 másodperc után?
> ysp = @(x) spline(T,Y,x);
```

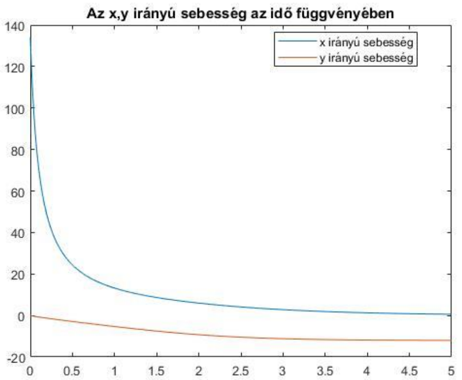
```
> ysp(2.25) % -12.7098 y irányú elmozdulás 2.25 másodperc után
```

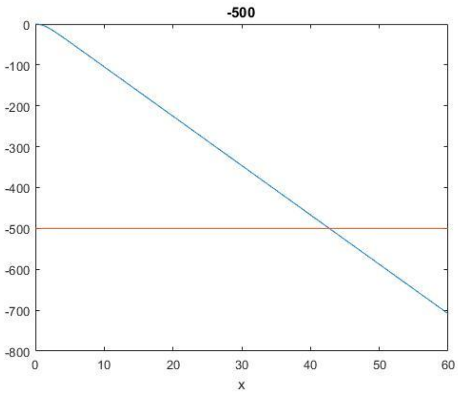
f) A függőleges elmozdulás: (2000 m-ről 1500 m-re): y=-500 m. Ehhez számoljuk ki hosszabb intervallumon, mondjuk 1 perc esetén, az elmozdulás értékeket!

```
> % A függőleges elmozdulás mikor lesz -500 m?
> [T,W]=ode45(ernyodiff,[0,60],[x0; y0; vx0; vy0])
> X = W(:,1); Y = W(:,2); % pozíciók
> VX = W(:,3); VY = W(:,4); % sebességek
> Y(end) %  -708.4927 m, y pozíció 1 perc után
> figure(4); plot(T,Y);
> ysp = @(t) spline(T,Y,t);
> hold on; plot(xlim,[-500 -500])
> % ysp(t) = -500 megoldása
> h = @(t) ysp(t) + 500;
> nyitas = fzero(h,40) %  42.7377 s
```









A FEJEZETBEN HASZNÁLT ÚJ FÜGGVÉNYEK

ode45	-	Közönséges differenciálegyenlet rendszer kezdeti érték problémájának megoldása Runge-Kutta módszerrel
odeset	-	Közönséges differenciálegyenlet kezdeti érték feladatát megoldó függvények opcionális paramétereinek megadása (pl. RelTol, AbsTol, MaxStep, InitialStep)
deval	-	Struktúra típusként megadott differenciálegyenlet megoldás értékének kiszámítása tetszőleges helyen
dsolve	-	Differenciálegyenlet szimbolikus megoldása