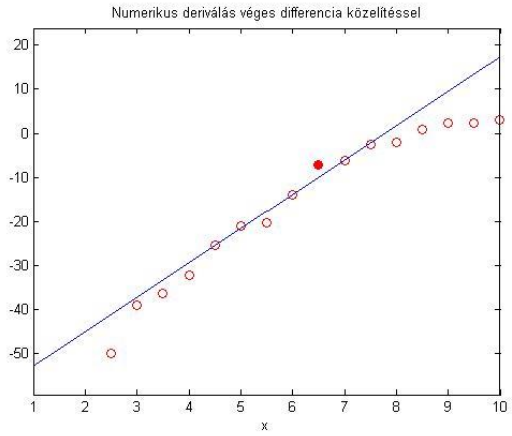
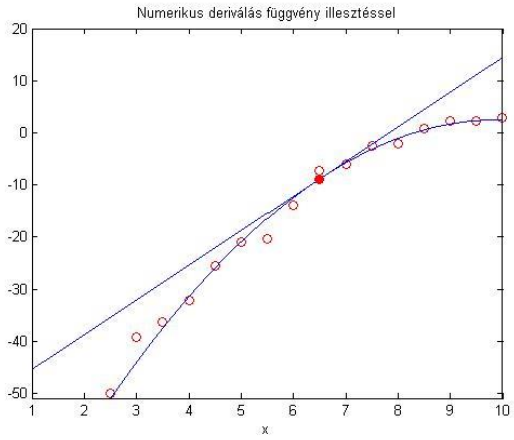
NUMERIKUS DERIVÁLÁS

A deriválttal (vagy differenciálhányadossal) egy mennyiség megváltozásának az ütemét tudjuk jellemezni. Nagyon sok helyen találkozhatunk vele a mérnöki gyakorlatban, tudományokban. Talán a legismertebb fizikai példa az út, sebesség gyorsulás összefüggése. Ha a megtett utat, pozíciót (x) idő függvényében (t) írjuk fel: $x = f(t)$, akkor a tárgy sebessége $v(t)$ az út idő szerinti első deriváltja lesz (az idő-út grafikon meredeksége): $v = \frac{df(t)}{dt}$, a gyorsulás pedig az út idő szerinti második deriváltja, vagy a sebesség első deriváltja: $a = \frac{dv(t)}{dt}$. Ugyancsak a deriváltat használhatjuk egy függvény minimumának vagy maximumának megkeresésére, mint azt láttuk korábban.

A deriválandó függvény lehet analitikus kifejezés vagy diszkrét pontokban megadott értékek. Egyszerű matematikai kifejezéssel megadott függvény deriváltja számítható analitikusan. Bonyolult matematikai kifejezések, vagy diszkrét pontok esetében azonban numerikus deriválást szoktak alkalmazni. Ugyancsak fontos szerepe van a numerikus deriválásnak a differenciál egyenletek megoldásakor.

A numerikus differenciálás egyik fontos módszere a derivált közelítése véges differencia hányadossal. Egy másik megközelítés, amikor a pontokat közelítjük valamilyen analitikusan felírható függvénnyel (pl. polinomiális regresszió, interpoláció) és az analitikus függvényt deriváljuk.





VÉGES DIFFERENCIA KÖZELÍTÉS

Tegyük fel, hogy csak diszkrét pontokban ismerjük a deriválandó függvény értékeit. A deriváltat közelíthetjük a szomszédos pontokat összekötő egyenes meredekségével. Erre több lehetőség is adódik. Az egyik a **jobb oldali vagy előremutató differencia**:

$$f'(x_i) = y_i' \approx \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

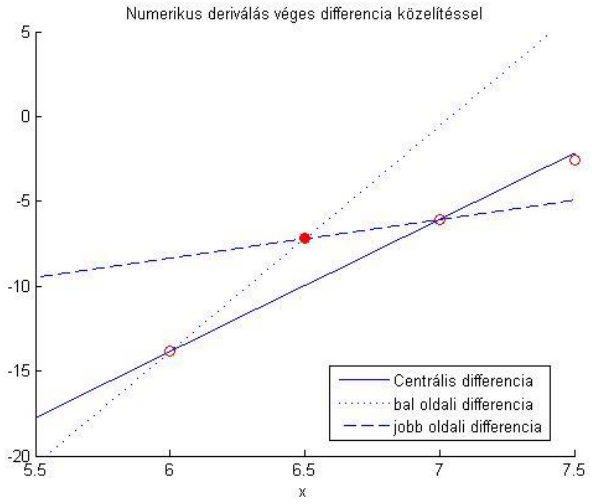
Egy másik lehetőség a **baloldali vagy hátramutató differencia**:

$$f'(x_i) = y_i' \approx \frac{y_i - y_{i-1}}{x_i - x_{i-1}}$$

Miután a jobb és baloldali differenciák hibáinak előjele gyakran ellentétes, jobb megoldást adhat a kettő átlaga. Amennyiben a pontok felosztása egyenletes ($x_{i+1} - x_i = x_i - x_{i-1} = h$) ez a **centrális differencia** formulához vezet:

$$f'(x_i) = y_i' \approx \frac{y_{i+1} - y_{i-1}}{x_{i+1} - x_{i-1}} = \frac{y_{i+1} - y_{i-1}}{2h}$$

Általában a centrális differencia formula pontosabb közelítését adja a deriváltnak, mint a jobb vagy baloldali differencia. A pontok közötti távolság csökkenése is pontosabb eredményhez vezet.



A VÉGES DIFFERENCIA KÖZELÍTÉSEK HIBÁI

A Taylor sorokat használhatjuk a jobb, bal és centrális differenciák csonkítási hibabecsléséhez.

$$f(x+h) = f(x) + h\,f'(x) + \frac{h^2}{2} f"(c)$$

ahol *c* egy ismeretlen szám *x* és *x+h* között. Legyen $x = x_i,\ x + h = x_{i+1}$ és fejezzük ki f'-t az egyenletből!

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h} - \frac{h}{2} f"(c)$$

A fenti formula a jobboldali (előremutató) differencia képlete, amelynek a hibája $-\frac{h}{2} f"(c)$, vagyis a csonkítási hiba nagyságrendje $O(h)$ (nagy ordó h). A fenti képletben h-t $(-h)$-ra cserélve megkapjuk a baloldali differencia képletét. A centrális differencia formula hiba becslését a harmadrendű Taylor sorból kaphatjuk meg:

$$f(x_{i+1}) = f(x_i + h) = f(x_i) + h\,f'(x_i) + \frac{h^2}{2} f''(x_i) + \frac{h^3}{3!} f'''(c_1)$$

$$f(x_{i-1}) = f(x_i - h) = f(x_i) - h\,f'(x_i) + \frac{h^2}{2} f''(x_i) - \frac{h^3}{3!} f'''(c_2)$$

ahol $x_i < c_1 < x_{i+1}$ és $x_{i-1} < c_2 < x_i$. Vonjuk ki egymásból a két egyenletet és fejezzük ki f'-t!

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2\,h} - \frac{h^2}{3!}\,\frac{f'''(c_1) + f'''(c_2)}{2}$$

A fentiek alapján a centrális differencia formula $O(h^2)$ nagyságrendű, ami sokkal jobb közelítést eredményez, így amikor csak lehet célszerű ezt használni.

Több pontra is felírhatjuk a deriváltat a pontosabb közelítés érdekében, például 5 pont bevonásával a centrális differencia formula hibája $O(h^4)$ lesz.

$$f'(x_i) = y_i' = \frac{y_{i-2} - 8\,y_{i-1} + 8\,y_{i+1} - y_{i+2}}{12\,h} + O(h^4)$$

A jobb és baloldali differencia hibája is csökkenthető több pont bevonásával. Az első derivált esetén a jobb és bal oldali differencia 3 pontra felírva:

$$f'(x_i) = y_i' \approx \frac{-3y_i + 4\,y_{i+1} - y_{i+2}}{2\,h} + O(h^2)$$

$$f'(x_i) = y_i' \approx \frac{y_{i-2} - 4y_{i-1} + 3y_i}{2\,h} + O(h^2)$$

MAGASABB RENDŰ DIFFERENCIA HÁNYADOSOK[1]

Magasabb rendű deriváltakra is felírható centrális differencia formula, ezeknél mind $O(h^2)$ lesz a hiba nagyságrendje. Centrális differencia formula második deriváltra 3 pontra:

$$f''(x_i) = y_i'' \approx \frac{y_{i+1} - 2\,y_i + y_{i-1}}{h^2} + O(h^2)$$

Centrális differencia formula harmadik deriváltra 4 pontra:

$$f'''(x_i) = y_i''' \approx \frac{1}{2\,h^3}(y_{i+2} - 2\,y_{i+1} + 2y_{i-1} - y_{i-2}) + O(h^2)$$

Centrális differencia formula negyedik deriváltra 5 pontra:

$$f^{(4)}(x_i) = y_i{}^{(4)} \approx \frac{1}{h^4}(y_{i+2} - 4\,y_{i+1} + 6y_i - 4y_{i-1} + y_{i-2}) + O(h^2)$$

A második derivált esetén a jobb és baloldali differencia 3 pontra:

$$f''(x_i) = y_i'' \approx \frac{y_i - 2\,y_{i+1} + y_{i+2}}{h^2} + O(h)$$

$$f''(x_i) = y_i'' \approx \frac{y_{i-2} - 2\,y_{i-1} + y_i}{h^2} + O(h)$$

A második derivált esetén a jobb és baloldali differencia 4 pontra:

$$f''(x_i) = y_i'' \approx \frac{2y_i - 5\,y_{i+1} + 4\,y_{i+2} - y_{i+3}}{h^2} + O(h^2)$$

$$f''(x_i) = y_i'' \approx \frac{-y_{i-3} + 4y_{i-2} - 5\,y_{i-1} + 2y_i}{h^2} + O(h^2)$$

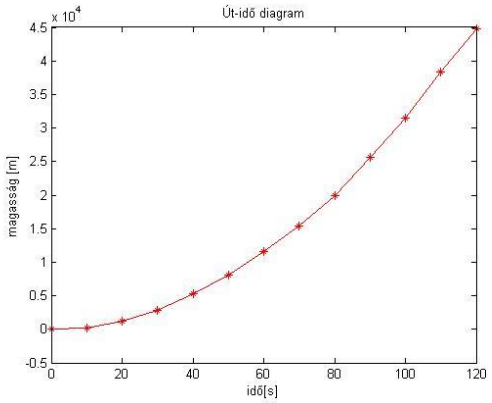
[1] Otthoni átnézésre

DIFFERENCIA HÁNYADOSOK ALKALMAZÁSA

Adottak egy űrsikló helyzetének magasság adatai a kilövés utáni első két percben:

t(s)	0	10	20	30	40	50	60	70	80	90	100	110	120
h(m)	-8	241	1244	2872	5377	8130	11617	15380	19872	25608	31412	38309	44726

Határozzuk meg az űrsikló sebességét az idő függvényében! Ahol lehet használjunk centrális differencia formulát! Töltsük be az adatokat az ursiklo.txt fájlból és ábrázoljuk a magassági pozíciót a idő függvényében!



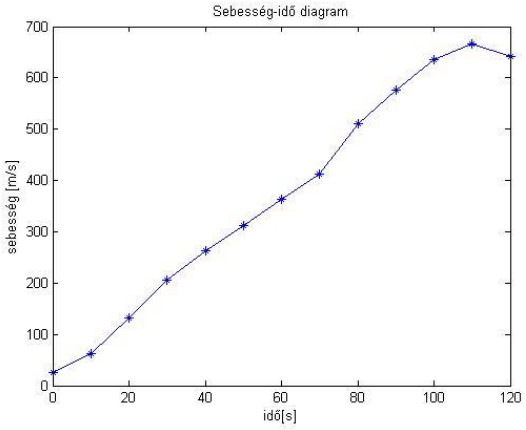

```
clc; close all; clear all;
adat = load('ursiklo.txt')
t = adat(:,1);x = adat(:,2);
figure(1); plot(t,h,'r*-')
xlabel('idő[s]')
ylabel('magasság [m]')
title('Út-idő diagram')
```

A sebesség az idő szerinti első derivált. Centrális formulát csak a második ponttól az utolsó előtti pontig tudunk használni. Az első pontban csak jobboldali (előremutató) differencia formulát, az utolsó pontban pedig csak baloldali (hátramutató) formulát használhatunk. Írjunk egy függvényt, ami kiszámolja minden pontban az első deriváltat, ahol lehet centrális differencia formulát alkalmazva!

```
function dx = derivalt(x,y)
% Numerikus deriválás differencia hányadosok alkalmazásával
    n = length(x);
    dx(1) = (y(2)-y(1))/(x(2)-x(1)); % jobboldali differencia
    for i = 2:n-1
       dx(i) = (y(i+1)-y(i-1))/(x(i+1)-x(i-1)); % centrális diff.
    end
    dx(n) = (y(n)-y(n-1))/(x(n)-x(n-1)); % baloldali differencia
end
```

Számoljuk ki az első deriváltat a fenti függvényt használva!

```
v = derivalt(t,x)
figure(2); plot(t,v,'b*-')
xlabel('idő[s]');
ylabel('sebesség [m/s]')
title('Sebesség-idő diagram')
```

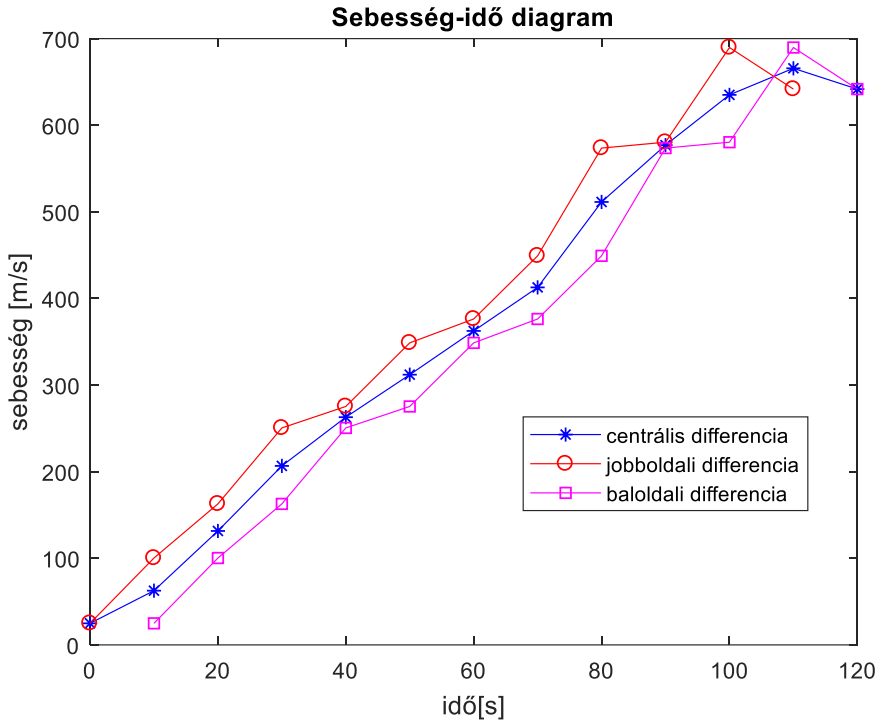


A Matlab beépített függvénye a **diff** is használható derivált számítására, mind numerikus, mind szimbolikus esetben. A **diff** parancsot numerikusan használva azonban csak egyoldali differenciákat számolhatunk vele, centrális differenciákat nem, ugyanis a **diff** parancs csak annyit tesz, ha egy vektor elemire meghívjuk, hogy kiszámolja a szomszédos elemek különbségét. Az eredménynek eggyel kevesebb eleme lesz, mint a bemenetnek volt. Határozzuk meg

a sebességeket és gyorsulásokat jobb és baloldali differencia különbségeket használva a **diff** paranccsal!

```
> % egyoldali differenciával beépített függvénnyel (diff)
> dx = diff(x);
> dt = diff(t);
> dxdt = dx./dt
> figure(2); hold on
> plot(t(1:end-1),dxdt,'ro-') % jobboldali/előremutató differencia
> plot(t(2:end),dxdt,'ms-') % baloldali/hátramutató differencia
> legend('centrális differencia','jobboldali differencia',...
>     'baloldali differencia','Location','best')
```

Az előremutató (jobboldali) és a hátramutató (baloldali) differenciák számítása lényegében megegyezik, jobboldali differenciákat az első ponttól az utolsó előttiig tudunk számítani, baloldalit pedig a 2. ponttól az utolsóig. A megjelenítésnél láthatjuk a különbséget, az értékek megegyeznek, csak eggyel el vannak csúsztatva a két esetben. Az ábra alapján látható, hogy a centrális differenciák használata simább görbét eredményezett, mint az egyoldali differenciáké. Hasonlóképp számíthatnánk a gyorsulásokat is a sebesség deriválásával!

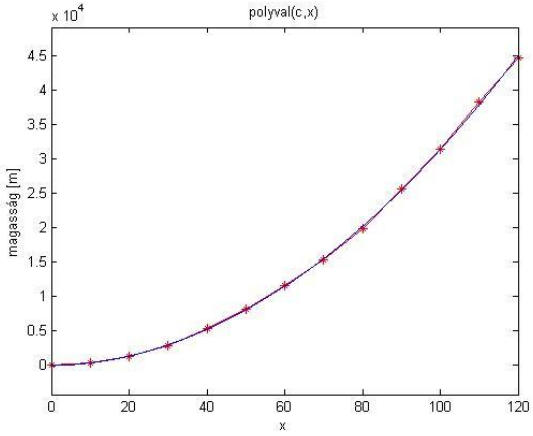


Az alkalmazott **deriált.m** függvény a 2. ponttól az utolsó előttiig alkalmaz $O(h^2)$ hibájú centrális differencia formulát, a két szélső pontban pedig $O(h)$ hibájú egyoldali differencia formulát (az ábrán látjuk, hogy a két szélső pontban az értéke megegyezik a jobb ill. baloldali formuláéval). Lehetne pontosítani a végpontokban is, $O(h^2)$ hibájú megoldást alkalmazva 3-3 pont bevonásával, a korábban ismertetett képletek alapján:

```
> % Az első és az utolsó pontban is O(h^2) hibájú közelítést alkalmazva
> n=numel(v)
> v1 = (-3*x(1) + 4*x(2)-x(3))/(t(3)-t(1)) %   -12.8000
> vn = (x(n-2) - 4*x(n-1) + 3*x(n))/(t(n)-t(n-2)) % 617.700
```

NUMERIKUS DERIVÁLÁS FÜGGVÉNYILLESZTÉSSEL

Nézzük meg a másik megközelítést is, amikor a pontokra egy közelítő függvényt illesztünk. Legyen most ez egy másodfokú polinom!

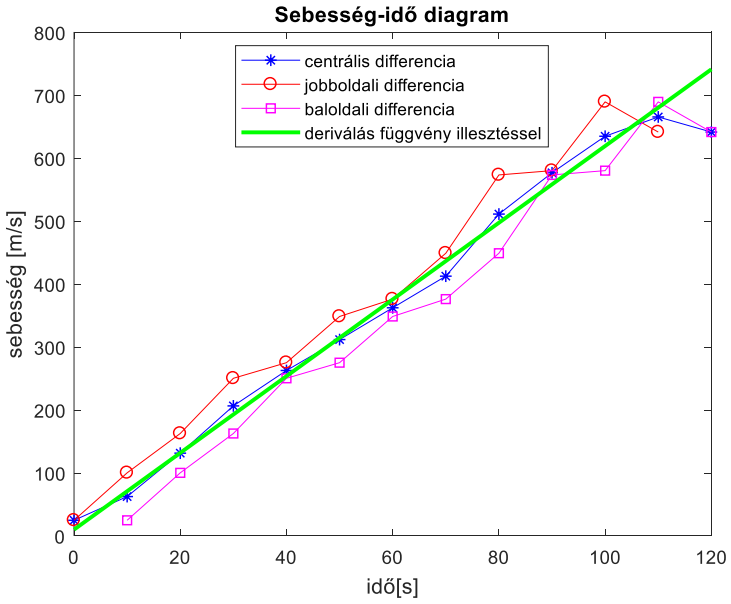



```
> % deriválás függvény illesztéssel
> c = polyfit(t,x,2)
> % c = 3.0470   10.1151  -89.3626
> p = @(x) polyval(c,x)
> figure(1); hold on;
> fplot(p,[min(t),max(t)]);
```

Az ábra alapján látjuk, hogy nagyon jól illeszkedik a pontokra a parabola. Ha az együtthatókkal definiálnánk szimbolikus alakban a polinomot (**sym**), akkor deriválásra használhatnánk a **diff** parancsot, mint azt már korábban láttuk. Azonban algebrai polinomok esetében van egy egyszerűbb megoldás is. A **polyder** parancsot használva nincs szükség szimbolikus formába alakítani a megoldást. Nézzük meg mindkét megoldást!

```
> % sebesség szimbolikus deriválással
> syms x
> ps = c(1)*x.^2 + c(2)*x + c(3)
> % (3050*x^2)/1001 + (50626*x)/5005 - 6288336567612965/70368744177664
> v2 = diff(ps,x)
> % (6100*x)/1001 + 50626/5005
> v2 = matlabFunction(v2)
> % vagy egyszerűbben polinom deriválás beépített Matlab paranccsal
> c1 = polyder(c) % c1 = 6.0939   10.1151
> v2 = @(x) polyval(c1,x)
> figure(2)
> fplot(v2,[min(t),max(t)],'g','LineWidth',2)
> legend('centrális differencia','jobboldali differencia',...
>     'baloldali differencia','deriválás függvény
illesztéssel','Location','best')
```

A derivált ebben az esetben még simább lefutású lett, egy egyenes!



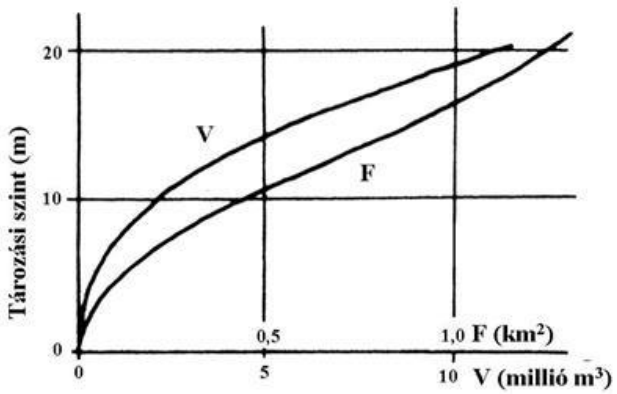
ÖNTÖZŐVÍZ TÁROLÓ PÁROLGÁSI, SZIVÁRGÁSI VESZTESÉGEI

Nézzünk meg egy jellegzetes építőmérnöki feladatot numerikus deriválásra! Becsüljük meg, hogy egy öntözővíz-tározónak egy hónap során hogyan alakultak a párolgási és elszivárgási veszteségei ($Q_{loss}(t)$), ha ismert a hozzáfolyás ($Q_{in}(t)$) és a vízkivétel ($Q_{out}(t)$) vízhozamának idősora, a vízállás idősora ($z(t)$), valamint a tározó térfogati jelleggörbéje. Az alapegyenletet a következő: a hozzáfolyásból kivonva a vízkivételt és a párolgási/elszivárgási veszteséget megkapjuk a térfogat változását:

$$Q_{in}(t) - Q_{out}(t) - Q_{loss}(t) = A(z)\frac{\mathrm{d}z}{\mathrm{d}t}$$

ahol $\frac{\mathrm{d}z}{\mathrm{d}t}$ a vízszint megváltozása, $A(z)$ pedig a tározó adott vízszinthez tartozó felszíne, amit a tározó jelleggörbéjéből határozhatunk meg.

Az ábrán egy minta tározó jelleggörbéje látható, amiről leolvasható, hogy egy adott vízszint (tározási szint) esetén mekkora lesz a tárolt víztérfogat, illetve a tározó nyílt vízfelszíne.

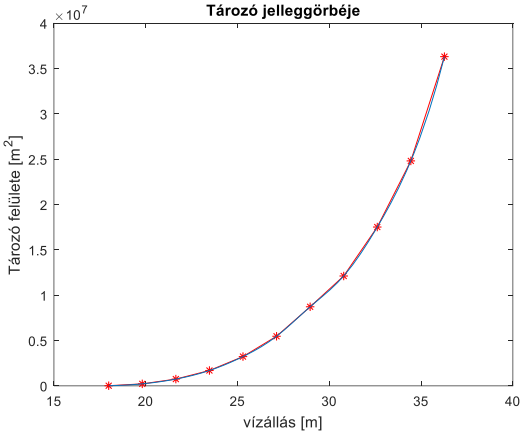


A kérdéses tározó esetén rendelkezésünkre áll a tározó jelleggörbéje a felszínre vonatkozóan, hogy adott vízszinthez [m] mekkora tározó felszín [m^2] tartozik.

vízszint *h* [m]	17.983	19.812	21.641	23.470	25.298	27.127	28.956	30.785	32.614	34.442	36.271
felszín *A* [m^2]	16177	222431	748178	1694521	3229775	5471806	8723345	12120476	17519486	24815707	36316941

A fenti adatok megtálalhatóak a **jelleggorbe.txt** fájlban is. Olvassuk be az adatokat, majd válasszuk szét a változókat. Tüntessük fel egy új ábrában a pontokat és illesszünk rá egy köbös másodrendű spline-t, függvény alakban! Az x tengelyen most a vízszint értékek legyenek.

```
> %% Víztározó
> clc; clear all; close all;
> % Jelleggörbe
> jelleg = load('jelleggorbe.txt');
> h = jelleg(:,1), A = jelleg(:,2),
> figure(1); plot(h,A,'r*-')
> F = @(x) spline(h,A,x)
> hold on; fplot(F, [min(h), max(h)])
> title('Tározó jelleggörbéje')
> xlabel('vízállás [m]');
> ylabel('Tározó felülete [m^2]')
```

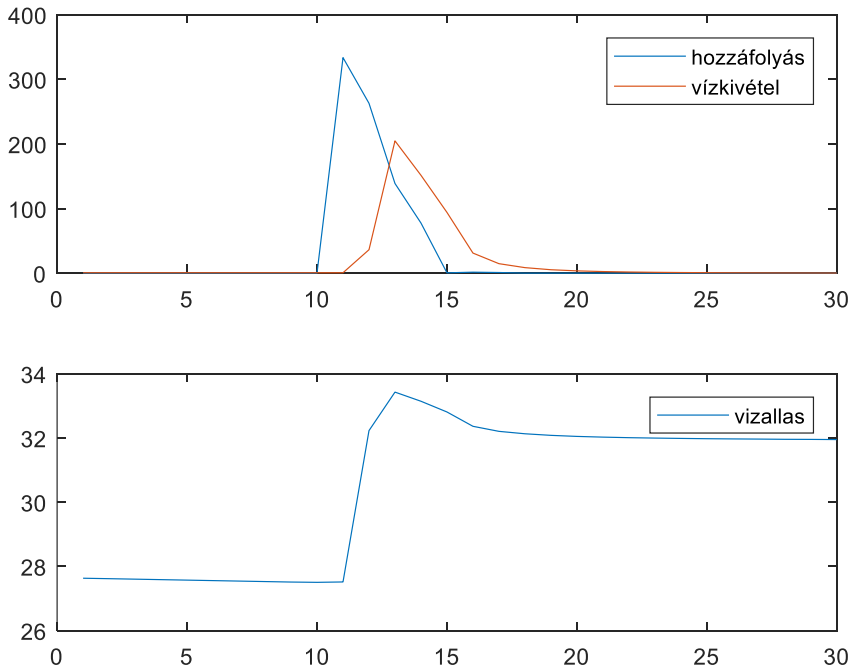


Rendelkezésünkre állnak mérési adatok 30 napra vízállásra ($z(t)$), a hozzáfolyás ($Q_{in}(t)$) és a vízkivétel ($Q_{out}(t)$) mennyiségére. Ezek az **idosor.txt** fájlban találhatóak, ahol négy oszlopban a következő adatok vannak: időpont - [nap], Q_{in}- [m^3/s], vízállás - [m], Q_{out} - [m^3/s].

időpont - t [nap]	hozzáfolyás - Q_{in} [m^3/s]	vízállás - z [m]	vízkivétel - Q_{out} [m^3/s]
1.	0.08	27.630	0.70
2.	0.08	27.616	0.70
⋮	⋮	⋮	⋮

Válasszuk szét az adatokat és a subplot parancsot használva készítsünk egy ábrát két egymás alatti grafikonnal, a felsőben feltüntetve a hozzáfolyás és a vízkivétel idősorát, az alsón pedig a vízszint megváltozását!

```
> % Idősor adatok
> idosor = load('idosor.txt');
> t=idosor(:,1), Qin=idosor(:,2),z=idosor(:,3), Qout=idosor(:,4)
> figure(2); subplot(2,1,1);
> plot(t,Qin,t,Qout); legend('hozzáfolyás','vízkivétel')
> subplot(2,1,2); plot(t,z); legend('vizallas')
```



A jelleggörbe alapján számoljuk ki az egyes vízállásokhoz tartozó felszín adatok idősorát $A(z)$!

```
> % Az adott idősorhoz tartozó tározó felszínek kiszámítása
> A = F(z)
```

Az alapegyenletünk a következő: $Q_{in}(t) - Q_{out}(t) - Q_{loss}(t) = A(z)\frac{\mathrm{dz}}{\mathrm{dt}}$.

Ebből ismerjük a $Q_{in}(t), Q_{out}(t)$, $A(z)$ idősorokat és a vízszint idősorát is $(z(t))$. Ahhoz, hogy ki tudjuk számolni a párolgási/elszivárgási veszteség idősorát $(Q_{loss}(t))$, a vízszint idő szerint változását vagyis az első deriváltat kellene meghatározzuk. Egy

adott napon mért hozzáfolyás, vízkivétel hatása mindig a másnap reggel mért vízszintekben fog csak megjelenni, ezért itt előremutató, vagy jobboldali differenciákra lesz szükségünk!

Határozzuk meg a vízállás idő szerinti első deriváltját ($\frac{dz}{dt}$), jobboldali (előremutató) differencia formulát használva ahol lehet (az utolsó pontnál baloldali differencia formulával)! Ehhez módosíthatjuk a korábban a centrális differencia formulára megírt **derivalt.m** függvényünket, vagy megoldhatjuk a Matlab numerikus deriválás parancsát alkalmazva is. Figyeljünk oda, hogy az idő mértékegységek megegyezzenek! A végeredmény a további számítások érdekében oszlopvektor legyen!

A jobb oldali vagy előremutató differencia képlete: $f'(x_i) = y_i' \approx \frac{y_{i+1}-y_i}{x_{i+1}-x_i}$

A baloldali vagy hátramutató differencia: $f'(x_i) = y_i' \approx \frac{y_i-y_{i-1}}{x_i-x_{i-1}}$

A módosított függvény neve legyen **jobbderivalt.m**!

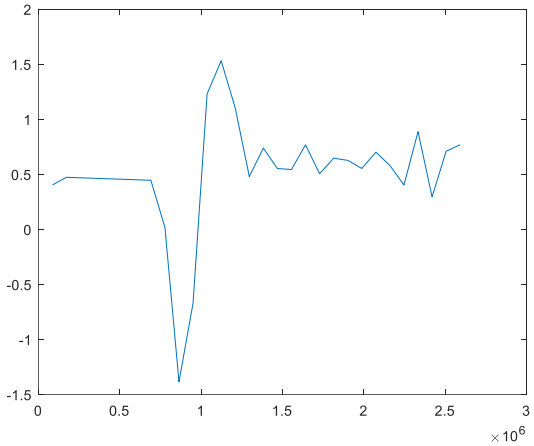
```
function dx = jobbderivalt(x,y)
% Numerikus deriválás differencia hányadosok alkalmazásával
% Utolsó előtti pontig jobb, utolsó pontban baloldali differencia
formulával
    n = length(x);
    for i = 1:n-1
       dx(i) = (y(i+1)-y(i))/(x(i+1)-x(i));
    end
    dx(n) = (y(n)-y(n-1))/(x(n)-x(n-1));
end
```

Számítsuk ki a vízállás változását a fenti függvénnyel! Ne felejtsük el a napokban megadott időket átváltani másodpercekre!

```
% vízállás változás
ts = t*24*3600;
dz = jobbderivalt(ts,z);
dz=dz'
% vagy beépített diff függvényt használva
dz2 = diff(z)./diff(ts); dz2 = [dz2; dz2(end)]
```

Számoljuk ki a tárolt víz térfogat változásának idősorát is: $\frac{dV}{dt} = A(z)\frac{dz}{dt}$, majd az alapegyenletet használva számolja ki a párolgási/szivárgási veszteség idősorát (Q_{loss}) [m^3/s]-ban!

```
% Térfogat változás, párolgási,
szivárgási veszteség [m^3/s]
dV = A.*dz
Qloss = Qin - Qout - dV;
figure(3); plot(ts,Qloss)
```

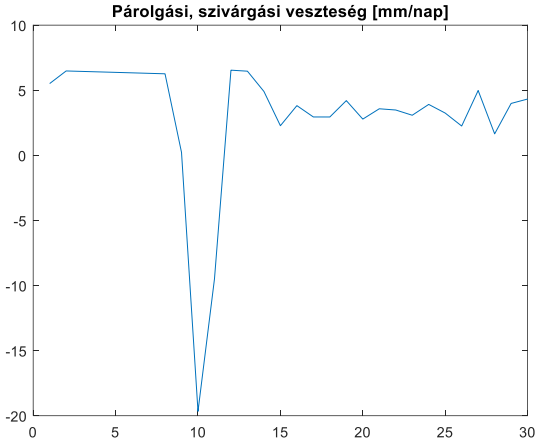


A párolgás a vízfelszínnel arányos (h_v), értékét mm/nap mennyiségben szokták megadni. Számoljuk át a [m^3/s]-ban megkapott értékeket mm/nap mennyiségre az alábbi képletet használva:

$$h_v = \frac{Q_{loss} \cdot 86400 \cdot 1000}{A}$$

ahol Q_{loss} [m^3/s]-ban adott, 86400 a másodpercek száma egy napban, A pedig az adott időponthoz tartozó vízfelszín [m^2]. Az 1000-es szorzó azért van benne, mert a végeredményt nem méterben, hanem mm-ben kapjuk (/nap). A kapott értékeket ábrázoljuk is! (Megj: A párolgási veszteség max. napi 10 mm szokott lenni, értéke lehet negatív is, csapadék esetén)

```
> % Párolgási, szivárgási veszteségek mm-ben
> veszt_mm = Qloss*86400*1000./A
> figure(3); plot(t,veszt_mm);
> title('Párolgási, szivárgási veszteség [mm/nap]')
```



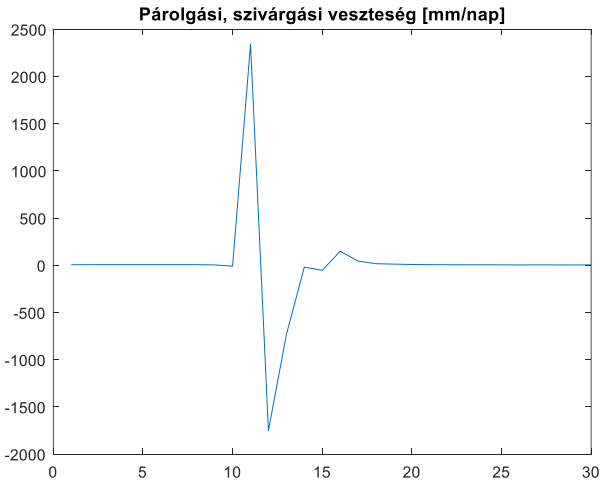
Mi történt volna, ha nem előremutató differencia formulákat használunk (aminek itt fizikai tartalma van), hanem itt is centrális differencia formulákat alkalmazzuk, mondván, hogy pontosabb? Cseréljük ki a

```
> dz = jobbderivalt(ts,z);
```

parancsot a következőre (ami centrális differencia formulát használ):

```
> dz = derivalt(ts,z);
```

Az eredmény szemmel láthatóan képtelenség, hiszen egy nap alatt nem szokott 2.3 m magasságú vízoszlop elpárologni, se 1.8 m csapadék esni.



DERIVÁLÁS TÖBBVÁLTOZÓS ESETBEN

A **gradiens** a deriválás általánosítása többváltozós esetre. Egy függvény deriváltjának értékét egyváltozós esetben számíthatjuk, értéke egy skalár lesz. Több változó esetén a gradiens veszi át a helyét, mely ellentétben a deriválttal, már vektort ad vissza, nem skalárt, eredménye egy vektormező lesz. Kétváltozós esetben egy *f* függvény gradiens vektora a következő:

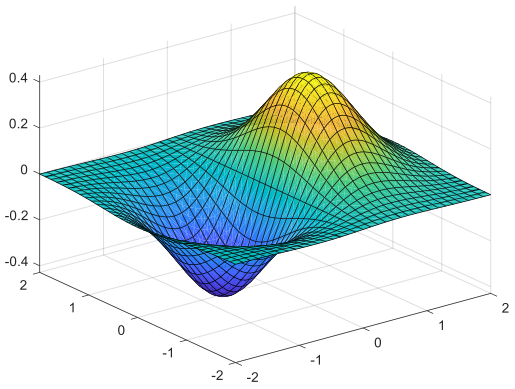
$$\nabla f(x,y) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix};$$

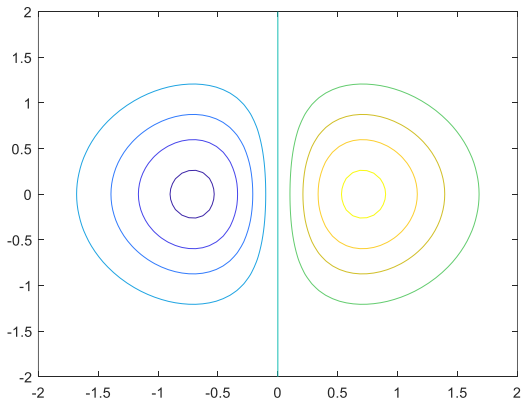
Ugyanúgy, ahogy a derivált az érintő meredekségét adja meg, a gradiens a felület legnagyobb meredekségű irányába mutat, megadja az adott pontban a függvény legnagyobb megváltozásának irányát, értékét.

Határozzuk meg a következő kétváltozós függvény gradiensét, és ábrázoljuk a vektormezőt az $x \in [-2,2]$ és $y \in [-2,2]$ tartományon!

$$f(x,y) = x\, e^{-(x^2+y^2)}$$

```
>  %% Kétváltozós felület definiálása
>  clc; clear all; close all;
>  f = @(x,y) x .* exp(-(x.^2 + y.^2));
>  figure(1)
>  fsurf(f,[-2,2,-2,2])
>  figure(2)
>  fcontour(f,[-2,2,-2,2])
```



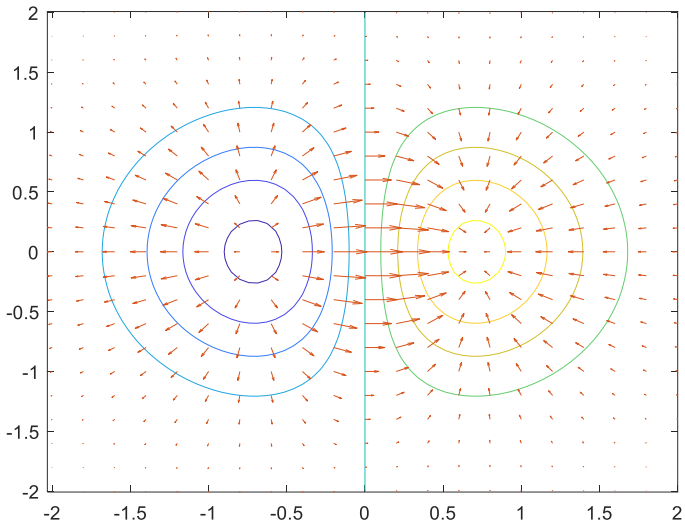


Számítsuk ki a gradiensvektor elemeit, vagyis az x és y szerinti parciális deriváltakat szimbolikusan! Ehhez használhatnánk szimbolikusan a **diff** parancsot is, vagy egy lépésben a **gradient** parancsot.

```
>  %% Gradiens számítás szimbolikusan
>  syms x y
>  fs = x .* exp(-(x.^2 + y.^2));
>  G = gradient(fs, [x, y])
>  %  exp(- x^2 - y^2) - 2*x^2*exp(- x^2 - y^2)
>  %                   -2*x*y*exp(- x^2 - y^2)
```

Ábrázoljuk a vektormezőt a szintvonalas ábrán, úgy, hogy kiszámoljuk egy rácsháló pontjaiban a gradiens értékeit! Vektormezőt a **quiver** paranccsal jeleníthetünk meg.

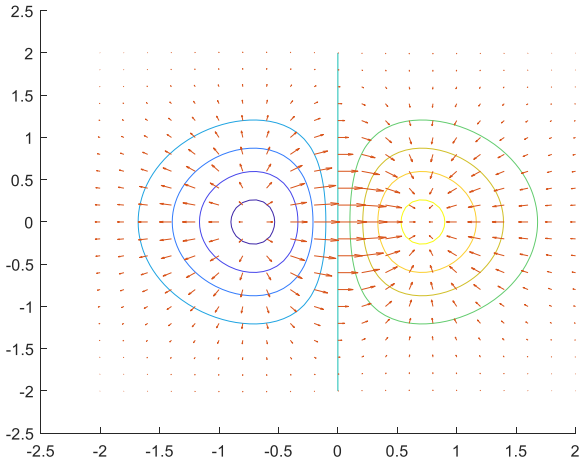
```
> [X, Y] = meshgrid(-2:0.2:2,-2:0.2:2);
> gx = matlabFunction(G(1))
> gy = matlabFunction(G(2))
> GX = gx(X,Y); GY = gy(X,Y);
> hold on;
> quiver(X,Y,GX,GY)
```



A **gradient** parancsot használhatjuk numerikusan is a gradiensvektor értékeinek kiszámítására.

```
> %% Gradiens számítás numerikusan
> Z = f(X,Y);
> [px,py] = gradient(Z);
> figure(3); hold on;
> fcontour(f,[-2,2,-2,2])
> quiver(X,Y,px,py)
```

Itt először kiszámítottuk az X,Y rácspontokban a függvény értékét (Z), majd egyenletesnek feltételezve a rácsháló osztását, a gradiens vektor értékeit (px,py).



A második parciális deriváltakat tartalmazza a Hesse mátrix, ami kétváltozós esetben így írható fel:

$$H(x,y) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x\,\partial y} \\ \frac{\partial^2 f}{\partial y\,\partial x} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix}$$

Egy *f* függvény *x* változó szerinti második parciális deriváltját számíthatjuk a **diff(f,x,2)** paranccsal, vagy az egész Hesse mátrixot is megadhatjuk egyben szimbolikusan a **hessian** parancs alkalmazásával. Határozzuk meg az előző függvény Hesse mátrixát és számoljuk ki a második deriváltak értékét az x=0.5 és y = 0.7 helyen!

```
> %% Hesse mátrix
> d2x = diff(fs,x,2)
> % 4*x^3*exp(- x^2 - y^2) - 6*x*exp(- x^2 - y^2)
> H = hessian(fs)
> % [   4*x^3*exp(- x^2 - y^2) - 6*x*exp(- x^2 - y^2), 4*x^2*y*exp(-
  x^2 - y^2) - 2*y*exp(- x^2 - y^2)]
> % [ 4*x^2*y*exp(- x^2 - y^2) - 2*y*exp(- x^2 - y^2), 4*x*y^2*exp(-
  x^2 - y^2) - 2*x*exp(- x^2 - y^2)]
> Hfv = matlabFunction(H)
> Hfv(0.5,0.7)
> %    -1.1928   -0.3340
> %    -0.3340   -0.0095
```

ÚJ FÜGGVÉNYEK A GYAKORLATON

polyder	-	Algebrai polinom deriváltjának számítása
diff(f,x,2)	-	f szimbolikus kifejezés 2. deriváltja x szerint
gradient	-	Gradiensek számítása numerikusan, szimbolikusan
quiver	-	vektormező megjelenítése
hessian	-	Hesse-mátrix, az f(x) függvény második parciális deriváltjainak a mátrixa