NUMERIKUS INTEGRÁLÁS

A numerikus integrálás egy közelítő integrálási módszert jelent, melynek alkalmazásával a határozott integrálok közelítő értékét adjuk meg. Integrálásra nagyon sok terület van szükség, legyen szó ívhossz számításról ($L = \int_a^b \sqrt{1 + (f'(x))^2}dx$), terület, térfogat számításról vagy differenciál egyenletek megoldásáról.

NUMERIKUS INTEGRÁLÁS TRAPÉZ SZABÁLYT ALKALMAZVA

Az egyik legismertebb módszer a trapéz-szabály alkalmazása, ahol a két szomszédos pontot összekötő egyenes alatti trapéz területével közelítjük az adott szakaszra az integrált. Ezt használhatjuk diszkrét pontok esetében is és analitikusan megadott függvények esetében is, amikor nem tudjuk meghatározni szimbolikusan az integrált. Ez utóbbi esetben felveszünk n pontot az integrálandó függvény szakaszon, n-1 szakaszra osztva a tartományt, és ezekre a pontokra alkalmazzuk a trapéz szabályt.

Egyetlen intervallum esetén:

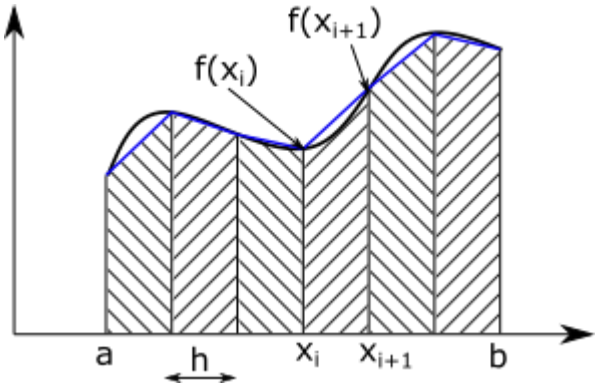
$$\int_a^b f(x)dx \approx \frac{f(a) + f(b)}{2}(b - a)$$

Több intervallumnál összegezni kell a trapézok területét. $N = n - 1$ részintervallum esetén:

$$\int_a^b f(x)dx \approx \frac{1}{2}\sum_{i=1}^{N}\big(f(x_i) + f(x_{i+1})\big)(x_{i+1} - x_i)$$

A fenti képletben nem szükséges, hogy az egyes részintervallumok azonos hosszúságúak legyenek, ha azonban az intervallumok hossza megegyezik $(x_2 - x_1) = (x_3 - x_2) = \cdots = (x_n - x_{n-1}) = h$, akkor egyszerűsíteni lehet a képletet:

$$\int_a^b f(x)dx \approx \frac{h}{2}\sum_{i=1}^{N}\big(f(x_i) + f(x_{i+1})\big)$$



Nézzük meg a fentieket egy példán keresztül!

A Föld sűrűsége (ρ) a sugarával (R) együtt változik, megközelítően az alábbiak szerint:

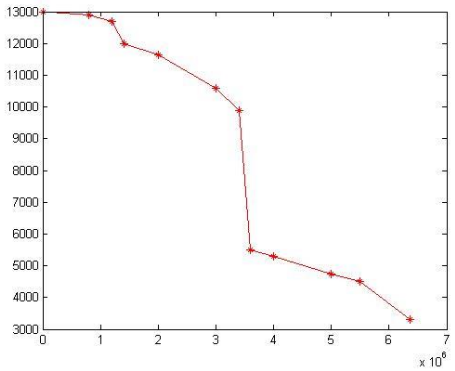
Sugár [km]	0	800	1200	1400	2000	3000	3400	3600	4000	5000	5500	6370
Sűrűség [kg/m^3]	13000	12900	12700	12000	11650	10600	9900	5500	5300	4750	4500	3300

Határozza meg a Föld tömegét az alábbi integrál alapján (a számításoknál figyeljünk a mértékegységekre)!

$$M_{Föld} = \int_0^{6370} \rho 4\pi R^2 dr$$

A sűrűségadatokat beolvashatjuk a **'fold_surusege.txt'** fájlból, vagy beírhatjuk kézzel is.

```
> % Föld tömege
> clc; close all; clear all;
> adat = load('fold_surusege.txt')
> R = adat(:,1)*1000
> ro = adat(:,2)
> figure(1); plot(R,ro,'m*-')
```



Számítsuk ki az integrálandó függvény értékeit a megadott sugár értékekhez!

```
> fx = 4*pi*ro.*R.^2
```

A megoldáshoz használjuk most a Matlab beépített **trapz** függvényét, ami diszkrét pontok alapján közelíti a határozott integrál értékét trapéz szabályt alkalmazva. A pontoknak nem kell egyenletesen elhelyezkedniük.

```
> M = trapz(R,fx) %  6.0261e+24 kg
```

Vagyis a Föld tömegére $6.0261 \cdot 10^{24}$ kilogrammot kaptunk. Ez nagyságrendileg stimmel, a jelenleg elfogadott becslés a Föld tömegére $(5.9722 \pm 0.0006) \cdot 10^{24}$ kg.

NUMERIKUS INTEGRÁLÁS SIMPSON-SZABÁLLYAL

A trapéz szabály egyenesekkel közelíti a szomszédos pontok között a függvényt. Pontosabb eredményt lehet elérni könnyen integrálható, magasabb rendű függvény közelítés alkalmazásával. A legismertebb ilyen módszerek a Simpson-formulák, amelyek másod vagy harmadfokú polinomokkal közelítik a függvény szakaszokat (Simpson's 1/3 method, Simpson's 3/8 method). A másodfokú Simpson-formula esetében 3 szomszédos pontra lehet illeszteni egy parabolát. Ezt megtehetjük a Newton-féle interpolációs polinomot felírva 3 pontra:

$$p(x) = a_1 + a_2(x - x_1) + a_3(x - x_1)(x - x_2)$$

Ahol az együtthatók a következők:

$$a_1 = y_1; a_2 = \frac{y_2 - y_1}{x_2 - x_1};\ a_3 = \frac{\frac{y_3 - y_2}{x_3 - x_2} - \frac{y_2 - y_1}{x_2 - x_1}}{x_3 - x_1}$$

Egyenlő (h) intervallumok esetén:

$$a_1 = y_1; a_2 = \frac{y_2 - y_1}{h};\ a_3 = \frac{y_3 - 2\,y_2 + y_1}{2\,h^2}$$

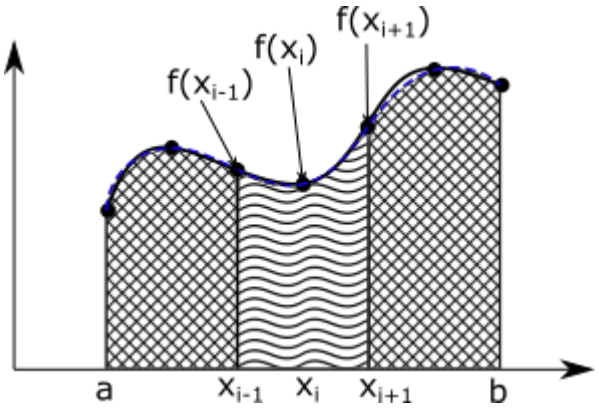
Visszahelyettesítve az együtthatókat a polinomba, levezethető a következő képlet 3 pontra:

$$\int_{x_1}^{x_3} f(x)dx \approx \int_{x_1}^{x_3} p(x)dx = \frac{h}{3}\left(f(x_1) + 4\,f(x_2) + f(x_3)\right)$$

Általánosítva:

$$\int_{x_{i-1}}^{x_{i+1}} f(x)dx \approx \frac{h}{3}\left(f(x_{i-1}) + 4\,f(x_i) + f(x_{i+1})\right)$$

Legyen n pontunk egyenletesen felvéve, egymástól h távolságra az [a,b] intervallumon, ekkor $x_1 = a,\ x_n = b$. Az n pont $N = n - 1$ szakaszra osztja az intervallumot. A Simpson szabályhoz 3 pont szükséges a parabola illesztéshez, mindig két egymást követő szakaszra számolhatjuk csak az integrált, ezért mindig páros számú szakaszra kell felbontanunk a függvényt a módszer alkalmazásához:



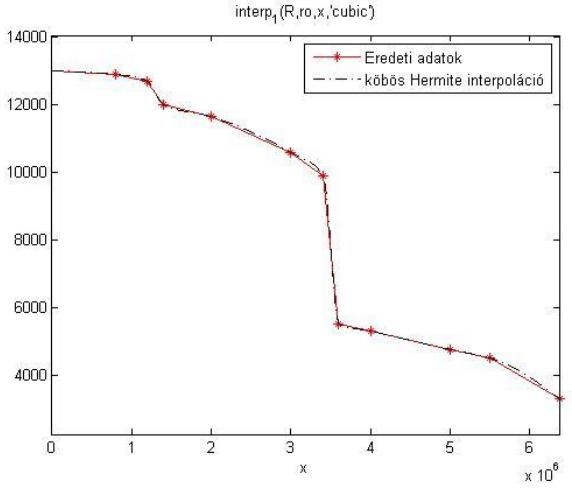
$$\int_a^b f(x)dx = \int_{x_1=a}^{x_3} f(x)dx + \int_{x_3}^{x_5} f(x)dx + \cdots \int_{x_{n-1}=x_N}^{x_n=b} f(x)dx = \sum_{i=2,4,6\ldots}^{N} \int_{x_{i-1}}^{x_{i+1}} f(x)dx$$

A három pontra kapott egyenletet felhasználva, n pontra a következő képletet kapjuk:

$$\int_a^b f(x)dx \approx \frac{h}{3}\left[f(a) + 4\sum_{i=2,4,6\ldots}^{n-1} f(x_i) + 2\sum_{j=3,5,7\ldots}^{n-2} f(x_j) + f(b)\right]$$

Számoljuk ki a Föld tömegét a Simpson szabályt alkalmazva is! Matlab-ban ezt a **quad** paranccsal tehetjük meg. A **quad** parancshoz azonban nem diszkrét pontokat, hanem egy függvényt kell megadnunk. Ehhez illesszünk a sűrűség adatokra egy spline görbét! Ugyanezt a példát néztük az interpoláció témakörénél is, ott láttuk, hogy a görbében lévő jelentős törések miatt a köbös másodrendű spline illesztés (**spline** parancs) nagy oszcillációt eredményez, ezért most használjunk a köbös elsőrendű interpolációt (**interp1** parancs **'pchip'** módszere).

```
> % köbös Hermite interpoláció
> ro_cubic = @(x) interp1(R,ro,x,'pchip')
> figure(1); hold on;
> g = fplot(ro_cubic, [0 6370000]);
> set(g,'Color', 'k','Linestyle','-.','LineWidth',1);
> legend('Eredeti adatok','köbös Hermite interpoláció')
```



Végezzük el az integrálást a Matlab Simpson-szabályt alkalmazó **quad** parancsával is! Ehhez adjuk meg az integrálandó mennyiséget a sugár függvényeként, felhasználva az előbb illesztett görbét! Majd számítsuk ki az integrált!

```
> fx_cubic = @(R) 4*pi*ro_cubic(R).*R.^2
> M2 = quad(fx_cubic,0,6370000) % 5.9658e+24 kg
```

A Föld tömegére most $5.9658 \cdot 10^{24}$ kilogrammot kaptunk. Ez jóval közelebb áll az elfogadott becsléshez, mint a trapéz szabállyal kapott érték.

Megjegyzés: A **quad** parancs használatát már nem javasolják a Matlab-on belül, későbbi verziókban már nem is lesz benne, hanem helyette az **integral** nevű parancs használatát ajánlják, ami komplexebb esetekben is jól működik. Ez már nem a hagyományosnak tekinthető Simpson szabályt, hanem adaptív kvadratúrát alkalmaz az integrál kiszámítására. A meghívása megegyezik a **quad** parancséval. Most ebben az esetben mégis a Simpson-módszerrel kiszámolt integrál értéke áll közelebb a ténylegeshez.

```
> M3 = integral(fx_cubic,0,6370000) % 6.0541e+24
```

Kiegészítés: Mind a numerikus deriválás, mind a numerikus integrálás pontosítására használható a Richardson-féle extrapoláció. Ezzel a módszerrel a csonkítási hibát tudjuk csökkenteni, úgy, hogy két pontatlanabb közelítést kombinálva egy nagyságrenddel pontosabb közelítést állíthatunk elő. Ennek a részleteivel most idő hiányában nem foglalkozunk.

TÖBBDIMENZIÓS INTEGRÁLOK SZÁMÍTÁSA SZABÁLYOS TARTOMÁNYON

Két és három dimenziós integrálok számítása is egy gyakran felmerülő feladat, ilyen például a terület, térfogat számítás is. Egy két dimenziós határozott integrál az alábbi alakba írható:

$$I = \int_{y_1}^{y_2} \int_{x_1}^{x_2} f(x,y)\, dx\, dy$$

Az integrálás ilyenkor két lépésre bontható, egy belső és egy külső integrálásra. Felírhatjuk először a külső integrált valamelyik korábbi módszerrel (trapéz, Simpson szabály), ahol minden egyes tag egy belső integrált fog tartalmazni, amiket szintén numerikusan számolhatunk. Így az egyváltozós numerikus integrálást lehet általánosítani több dimenzióra, szabályos (téglalap, téglatest) tartomány esetén.

Matlab-ban szabályos tartomány esetén egy függvény kettős integráljának kiszámítására használhatjuk az **integral2** parancsot, 3 dimenziós esetben pedig az **integral3** parancsot.

```
> q = integral2(fun,xmin,xmax,ymin,ymax)
> q = integral3(fun,xmin,xmax,ymin,ymax,zmin,zmax)
```

Az előbbire néztünk is már egy példát a 2D interpoláció témakörében, egy szabályos rácshálóban megadott terep térfogatának számításánál. Egészen más megközelítést kell alkalmaznunk azonban, ha az integrálási tartomány szabálytalan alakú.

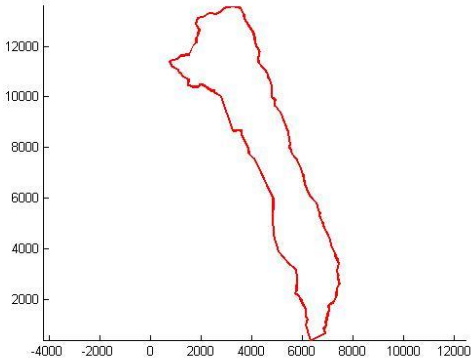
TÖBBDIMENZIÓS INTEGRÁLOK SZÁMÍTÁSA SZABÁLYTALAN TARTOMÁNYON

Szabálytalan tartományon az integráláshoz használhatjuk a **Monte-Carlo módszer**t. Ez egy sztochasztikus algoritmus, ami véletlen számokat használ. A hagyományos integrálási módszerek általában egy szabályos rácson értékelik ki az integrandust, míg ebben az esetben véletlen pontokban történik a függvény kiértékelés. Ezt a módszert a legkönnyebben terület, térfogat számításon lehet bemutatni, de a módszer általánosítható más esetekre is.

TERÜLET SZÁMÍTÁS MONTE-CARLO MÓDSZERREL

Meghatároztuk egy vízgyűjtő terület határának a pontjait, számítsuk ki a vízgyűjtő területét! Ehhez először töltsük be a **'vizgyujto.txt'** állományt, jelenítsük meg, a torzulások elkerülése végett azonos léptékkel az x és y tengelyen a határpontokat!

```
> clc; clear all; close all;
> adat = load('vizgyujto.txt');
> x = adat(:,1); y = adat(:,2);
> figure(1); hold on;
> plot(x,y,'r-','LineWidth',2)
> axis equal
```



Ha a területet Monte-Carlo módszerrel szeretnénk meghatározni, akkor az az alapelv, hogy meghatározzuk a terület befoglaló téglalapját és generálunk ezen a tartományon *N* véletlen pontot egyenletes eloszlásban. Ezután megszámoljuk, hogy hány pont esik az adott területre (*n*) és meghatározzuk azt az arányszámot (ρ), hogy a belül lévő pontok hogy viszonyulnak az összes ponthoz. Kellően sok pont felvétele esetén ez az arány közelítőleg a keresett terület és a befoglaló terület arányát adja:

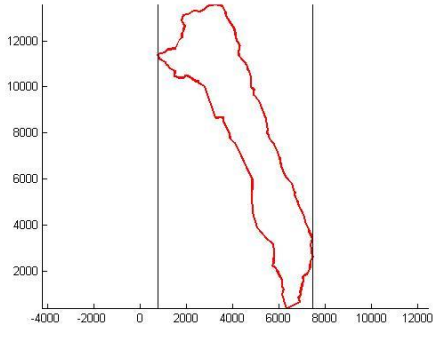
$$\rho = \frac{n}{N}$$

Ha ismerjük a befoglaló téglalap területét: $T = a \cdot b$, akkor a keresett szabálytalan tartomány (jelen esetben vízgyűjtő) területe T_v számítható:

$$T_v = \int_T f(x)\, dT \approx \rho \cdot T = \frac{n}{N} \cdot (a \cdot b)$$

Oldjuk meg ezt Matlab-ban! Először rajzoljuk meg a befoglaló téglalapot a területünk köré (**rectangle**)!

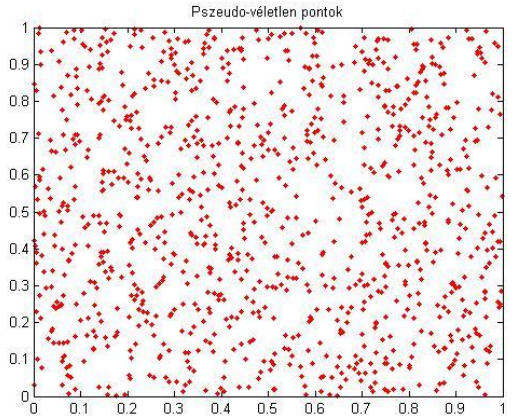
```
> a = max(x)-min(x) % 6.6698e+03
> b = max(y)-min(y) % 1.3169e+04
> rectangle('Position',[min(x),min(y),a,b])
```

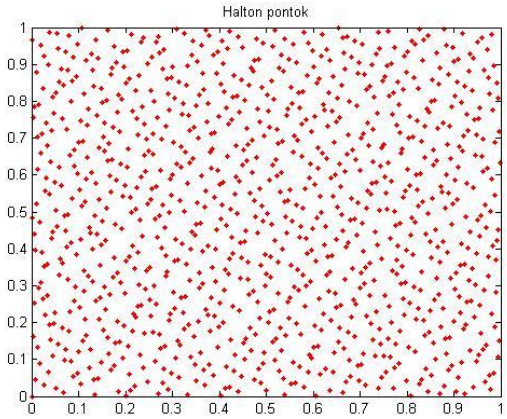


Generáljunk véletlen pontokat két dimenzióban! Ehhez több parancsot is használhatunk, az egyik a pszeudo véletlen számokat létrehozó **rand** parancs, a másik a Halton pontok (**haltonset**), amelyek a *van der Corput* sorozatokon alapulnak. Generáljunk velük 1000 pontot. Mindkettő a [0,1] tartományon dolgozik.

```
> % Pszeudo-véletlen pontok előállítása
> xyr = rand(1000,2);
```

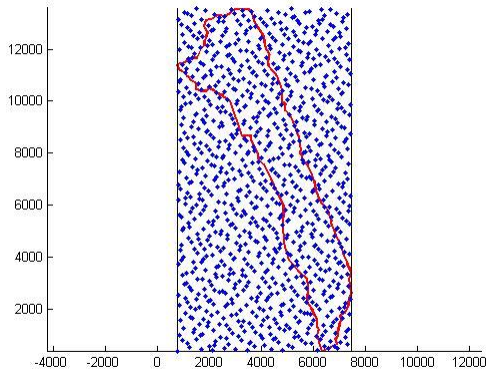
```
figure(2);
plot(xyr(:,1),xyr(:,2),'r.')
title('Pszeudo-véletlen pontok')

% Halton pontok előállítása
hs = haltonset(2); % 2 dimenziós Halton sorozat generálása
xyh = net(hs,1000);
figure(3);
plot(xyh(:,1),xyh(:,2),'r.')
title('Halton pontok')
```





A fenti ábrák közül az első a pszeudo véletlen pontokat, a jobb oldali a Halton pontokat mutatja. Látszódik, hogy ez utóbbi egyenletesebb eloszlású, így használjuk ezt a számításainkhoz!

Mivel a [0,1]x[0,1] tartományon vannak a pontjaink, transzformáljuk át őket a meghatározott befoglaló téglalap területére! Ehhez szorozzuk meg az oldalakat a megfelelő téglalap oldalhosszával és toljuk el a kezdőpontba!



```
% Halton pontok áttranszformálása
  a tartományba, ahol dolgozunk
xh = xyh(:,1)*a + min(x);
yh = xyh(:,2)*b + min(y);
figure(1);
plot(xh,yh,'b.')
```

A Monte-Carlo módszer alkalmazásához meg kell számoljuk, hogy hány pont van a tartományon belül. Ehhez a Matlab **inpolygon** parancsát használhatjuk, ami egy vektort ad vissza a belül lévő pontoknál egyesekkel, a többi pont indexénél nullákkal. A nem nulla elemeket megszámolhatjuk az **nnz** paranccsal (number of nonzero matrix elements).

```
% Terület számítás Monte-Carlo módszerrel
k = inpolygon(xh,yh,x,y);
plot(xh(k),yh(k),'r*')
n = nnz(k) % belül lévő pontok száma: 280
N = length(xh) % összes pont száma: 1000
T = a*b % teljes terület: 87824061 m^2
t = n/N*T % belül lévő terület: 2.4591e+07
format long
```

```
> t %  2.459073708000000e+07 = 24590737.08 m^2
```

Ellenőrzésképp kiszámíthatjuk a területet a geodéziában is használt, koordinátákra felírható trapézokra bontás módszerével is ($T = \sum \frac{(y_{i+1}+y_i)(x_{i+1}-x_i)}{2}$) !

```
> % Ellenőrzés: területszámítás koordináták alapján (trapéz módszer)
> x1 = x([2:end,1]); % eggyel balra tolt koordináták
> y1 = y([2:end,1]); % eggyel balra tolt koordináták
> Tp = sum((y1+y).*(x1-x)/2) %  24591531 m^2
```

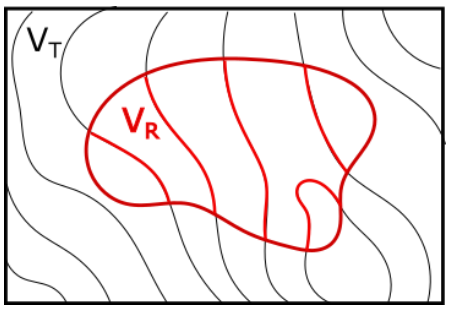
Vagy használható ugyanerre a Matlab beépített **trapz** parancsa is, ekkor az utolsó pont koordinátája után oda kell írjuk az első pont koordinátáit:

```
> x1 = [x; x(1)]; y1 = [y; y(1)];
> Tp = trapz(x1,y1) % 24591531
```

A két módszer hasonló eredményt adott, a Monte-Carlo módszer tovább pontosítható több pont felvételével. Egy szabálytalan idom területének kiszámolására sokféle módszer áll rendelkezésünkre, a Monte-Carlo módszer előnye abban nyilvánul meg, hogy általánosítható más esetekre is. Például ezzel a módszerrel kiszámolhatjuk, hogy mennyi csapadék hullott a vízgyűjtő területére, amennyiben ismerjük a csapadék eloszlását, például néhány helyen csapadékmérés történt!

MONTE-CARLO MÓDSZER ÁLTALÁNOSÍTÁSA

Fogalmazzuk meg a Monte-Carlo módszert általánosan! Legyen $f(x)$ értelmezve egy $x \in V_T$ tartományon és keressük a függvény határozott integrálját a tartomány egy V_R résztartományán $V_R \subset V_T$. a keresett integrál: $\int_{V_R} f(x)dV$.



A függvény átlagértékét a keresett tartományon kiszámíthatjuk az alábbi módon integrálással:

$$\bar{f_V} = \frac{1}{V_R} \int_{V_R} f(x)dV$$

Vagy felvehetünk a tartományon *n* pontot és kiszámolhatjuk ezekben a pontokban a függvény értékek átlagát, ami sok pont felvétele esetén közelítőleg megegyezik az integrálással kiszámolt értékkel:

$$\bar{f_V} \approx \frac{1}{n} \sum_{i=1}^{n} f(x_i)$$

ahol $x_i \in V_R$ és n a tartományba eső pontok száma.

Az átlagra kapott kifejezéseket egyenlővé téve kifejezhetjük az integrált:

$$\int_{V_R} f(x)dV \approx \frac{V_R}{n} \sum_{i=1}^{n} f(x_i)$$

A V_R tartomány közelítését megkaphatjuk, a területszámításnál is használt módon. Amennyiben a véletlenszerűen felvett pontok egyenletes eloszlást követnek, akkor

kellően sok pont esetén a tartományon belül lévő pontok száma úgy aránylik az összes ponthoz, mint a V_R rész tartomány nagysága a teljes V_T tartományhoz:

$$\frac{V_R}{V_T} = \frac{n}{N} \rightarrow V_R = \frac{V_T \cdot n}{N}$$

ahol n a tartományba eső, N pedig az összes pont száma. Az integrál közelítése tehát:

$$\int_V f(x)dV \approx \frac{\frac{V_T \cdot n}{N}}{n} \sum_{i=1}^{n} f(x_i) = \frac{V_T}{N} \sum_{i=1}^{n} f(x_i)$$

Vagyis az integrál számítható a tartományon belül lévő pontokban kiszámolt függvényértékek összegének és a (teljes tartomány nagysága/az összes pont) hányadosának szorzatával. A V_T/N tulajdonképpen az egy pontra jutó terület/térfogat nagyságát adja meg.

LEHULLOTT CSAPADÉKMENNYISÉG SZÁMÍTÁS MONTE-CARLO MÓDSZERREL

A megadott vízgyűjtő területen csapadékmérő állomásokat helyeztek el és megmérték a lehullott csapadék mennyiségét egy nagy vihar alatt, az állomásokon 5-12 mm esőt mértek helytől függően. Kérdés, hogy összesen mennyi csapadék hullott a vízgyűjtő terület egészére?

A csapadékmérő állomásokon mért csapadékmennyiségeket közelítették a következő másodfokú polinommal:

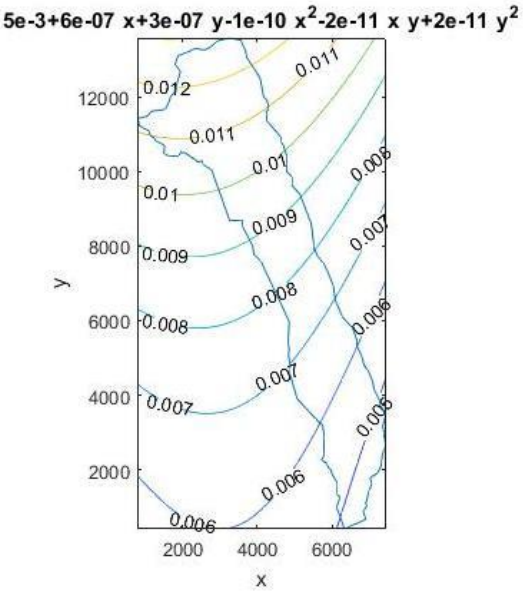
$$f(x,y) = 0.005 + 6 \cdot 10^{-7}\, x + 3 \cdot 10^{-7}\, y - 10^{-10}\, x^2 - 2 \cdot 10^{-11}\, xy + 2 \cdot 10^{-11}\, y^2$$

A fenti függvényt kellene integrálni a vízgyűjtő területére. Oldjuk meg a feladatot Monte-Carlo módszerrel!

Adjuk meg a fenti függvényt! Beírhatjuk kézzel is, vagy betölthetjük a csap.mat állományból!

```
> load csap.mat
> csap % @(x,y)5e-3+6e-07.*x+3e-07.*y-1e-10*x.^2-2e-11.*x.*y+2e-11*y.^2
> figure(4)
> h=ezcontour(csap,[min(x) max(x) min(y) max(y)])
```

```
> set(h,'Show','on'); hold on
> plot(x,y); axis equal;
```



Miután 1000 véletlen pontot már felvettünk a területen, használhatjuk itt is ezeket a pontokat. Az összes csapadékot megkapjuk, ha kiszámoljuk a területen belül lévő pontokban az átlagos csapadék mennyiséget, és ezt utána megszorozzuk a területtel (mivel ezt már az előbb meghatároztuk).

```
> xb = xh(k);
> yb = yh(k);
> n = length(xb) % 280
> N = length(xh) % 1000
> % Az átlagos csapadék a területen
> cs = 1/n*sum(csap(xb,yb)) %
  0.008612820224953 m
> % Az összes lehullott csapadék:
> CS = cs*t % 2.117955976691141e+05
```

Vagy használhatjuk az általánosított Monte-Carlo képletet, ez esetben nem kell kiszámolni a szabálytalan tartomány területét, elég csak a befoglaló téglalap területe és a felvett pontok száma, illetve a tartományon belül lévő pontokban a függvényértékek összege.

```
> % Az általános Monte-Carlo módszert használva,
> % ha a területet nem számoljuk ki külön
> CS2 = T/N*sum(csap(xb,yb)) % 2.117955976691141e+05
```

Amennyiben nem szabálytalan, hanem szabályos területen kell elvégezni az integrálást, ha például a befoglaló téglalapon lehullott csapadék mennyiségére vagyunk kíváncsiak, akkor használhatjuk az **integral2** parancsot. Az **integral2** parancshoz egy függvényt és a 2D tartomány határait kell megadjuk.

```
> % A befoglaló téglalapon leesett csapadék mennyisége
> CS_teglalap = integral2(csap,min(x),max(x),min(y),max(y))
> % 7.197677742886153e+05
```

GYAKORLÓ FELADAT NUMERIKUS DERIVÁLÁSHOZ, INTEGRÁLÁSHOZ[1]

Adott egy q [N/m] fajlagos súlyú szabadvezeték, amelyet két egymástól b távolságra lévő h magasságú oszlophoz rögzítünk úgy, hogy a feszítőerő vízszintes komponense H. A kábel alakját az alábbi koszinusz - hiperbolikus függvény írja le:

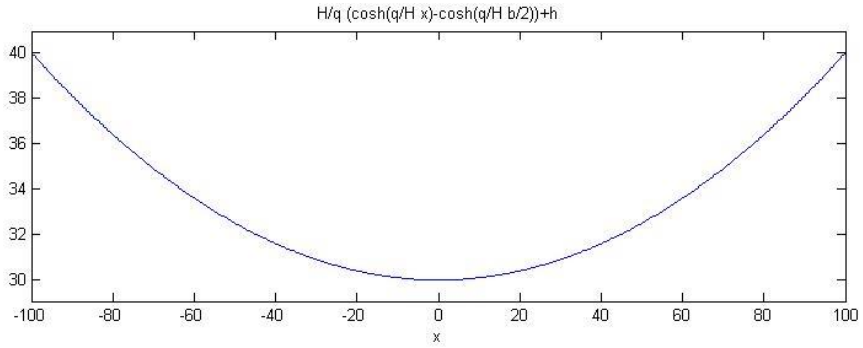
$$f(x,H,q,b,h) = \frac{H}{q} \cdot \left(cosh\left(\frac{q}{H} \cdot x\right) - cosh\left(\frac{q}{H} \cdot \frac{b}{2}\right) \right) + h$$

Határozzuk meg a kábel hosszát, ha

H = 1000 N, q = 2 N/m, b = 200m, h = 40m.

[1] Otthoni átnézésre!

Oldjuk meg a feladatot numerikusan és ellenőrizzük a hibáját szimbolikus számítással!



Egy f(x) függvény ívhosszát [a, b] intervallumon az alábbi módon határozhatjuk meg:

$$s = \int_a^b \sqrt{1 + (f'(x))^2}dx$$

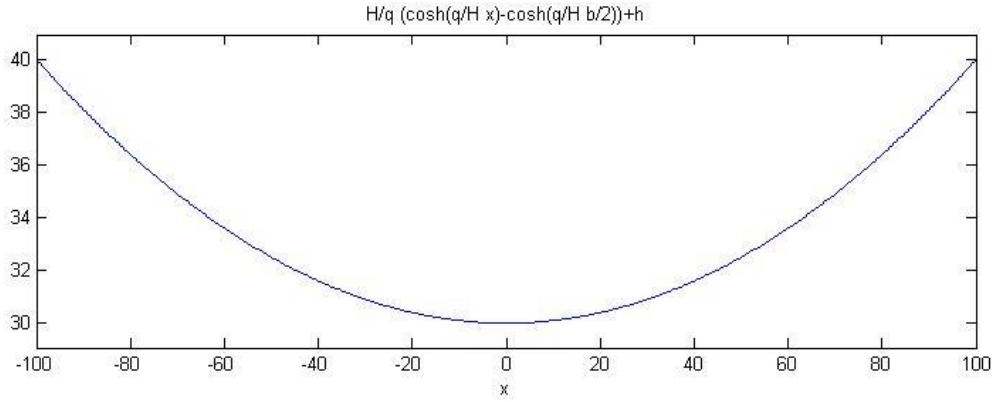
Oldjuk meg a következő feladatokat:

1. Állítsuk elő a pontos megoldást szimbolikusan a későbbi ellenőrzéshez!
2. A derivált függvényt közelítsük a másodrendű hibájú, $O(h^2)$ differencia módszerrel!

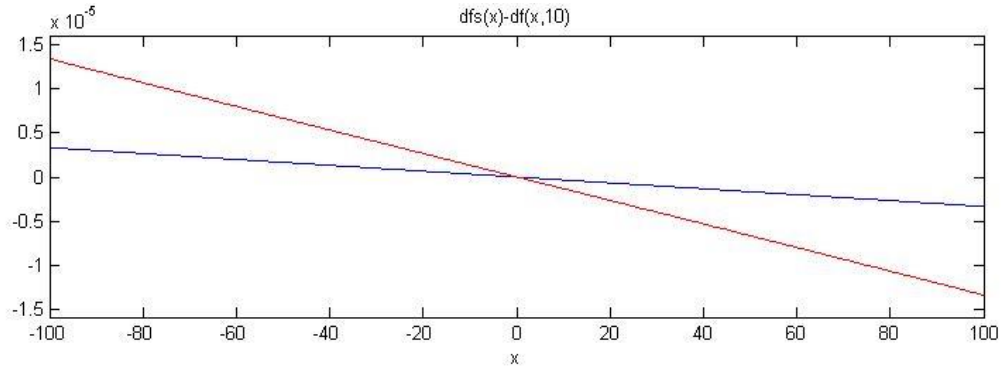
$$dF(x,\Delta) = \frac{F(x+\Delta) - F(x-\Delta)}{2 \cdot \Delta}$$

   Írjuk fel a derivált függvény közelítését Δ lépésköz függvényeként! Hasonlítsuk össze a pontos megoldással grafikusan $\Delta = 5$ és $\Delta = 10$ esetén!
3. Számítsuk ki az ívhosszat a trapézszabállyal $\Delta = 5$ és $\Delta = 10$ esetén!
4. Végezzük el az integrálást a negyedrendű hibájú Simpson szabály alapján is!
5. Ábrázoljuk a hibákat grafikusan oszlopdiagram segítségével!

```
> %% Ívhossz számítás
> % ívhossz: S=int(sqrt(1+df^2),a,b))
>
> clear all; format short; clc; close all;
>
> H=1000; % Feszítőerő vízszintes komponense [N]
> q=2;    % Szabadvezeték fajlagos súlya [N/m]
> b=200;  % Két oszlop közötti távolság [m]
> h=40;   % Oszlop magassága [m]
>
> f=@(x) H/q*(cosh(q/H*x)-cosh(q/H*b/2))+h;
> % A kábel alakját leíró függvény
>
> figure(1); clf;
> fplot(f, [-100,100])
```

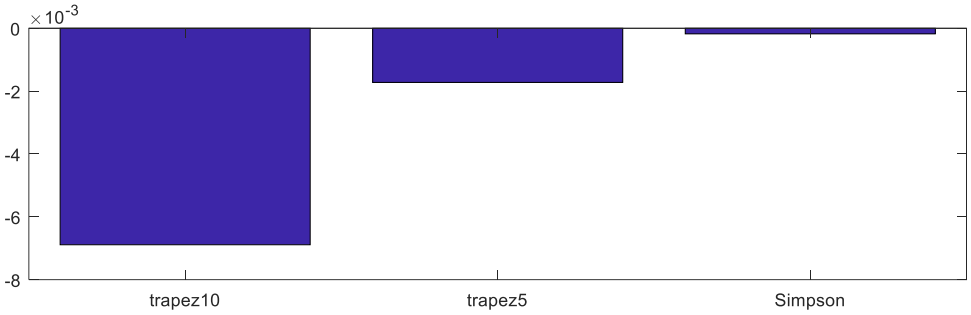



```
>  % A pontos megoldás - szimbolikusan
>  syms x;
>  fs =  H/q*(cosh(q/H*x)-cosh(q/H*b/2))+h
>  diff(fs,x)
>  % ans = sinh(x/500)
>  fsint = sqrt(1+sinh(x/500)^2)
>  ihsym = int(fsint,x,-100,100)
>  % ihsym = 500*exp(1/5) - 500/exp(1/5)
>  ihsym=double(ihsym)
>  % ihsym = 201.3360
>
>  % Numerikus megoldás
>  % A deriváltfüggvényt a másodrendű hibájú O(h^2) differencia
   módszerrel közelítjük
>  df=@(x,d) (f(x+d)-f(x-d))/(2*d); % numerikus közelítés
>  dfs=@(x) sinh(x/500); % a pontos megoldás szimbolikus számításból
>
>  % delta=5 és delta=10 esetén a deriváltak közelítésének hibája
>  d5=@(x) dfs(x)-df(x,5);
>  d10=@(x) dfs(x)-df(x,10);
>
>  % a hibafüggvények ábrázolása
>  figure(2);clf;
>  fplot(d5, [-100,100]);
>  hold on
>  fplot(d10, [-100,100]);
```




```
>  %% Integrálás - ívhossz: S=int(sqrt(1+df^2),a,b))
>  % A pontos érték: 201.3360
>
>  % Integrálás a trapéz szabály alapján (másodrendű hibával) -
   trapz(x,y)
```

```
> % x,y vektorok (összetartozó értékek)
> x=-100:10:100; % delta=10
> y10=sqrt(1+df(x,10).^2);
> ihtr10=trapz(x,y10)
> % ihtr10 = 201.3429
>
> x=-100:5:100; % delta=5
> y5=sqrt(1+df(x,5).^2);
> ihtr5=trapz(x,y5)
> % ihtr5 = 201.3377
>
> % Integrálás Simpson szabály alapján (negyedrendű hibával) -
  quad(fun,a,b)
> % fun-függvény, a,b-integrálási határok
> i10=@(x) sqrt(1+df(x,10).^2); % integrálandó függvény
> ihSimp=quad(i10,-100,100)
> % ihSimp = 201.3362
>
> % Hibák
> etr10=ihsym-ihtr10 % etr10 = -0.0069, Trapézszabály hibája, delta=10
> etr5=ihsym-ihtr5   % etr5  = -0.0017, Trapézszabály hibája, delta=5
> eSimp=ihsym-ihSimp % eSimp = -1.7709e-004, Simpson módszer hibája
> figure(3)
> bar([etr10 etr5 eSimp])
> names = {'trapez10'; 'trapez5'; 'Simpson' };
> set(gca,'XTickLabel',names)
```



ÚJ FÜGGVÉNYEK A GYAKORLATON

trapz(x,y)	-	Numerikus integrálás diszkrét pontok alapján trapéz szabállyal
quad(fun,a,b)	-	Függvény numerikus integrálása Simpson-szabállyal
integral(fun,a,b)	-	Függvény numerikus integrálása adaptív kvadratúrával
integral2	-	Kettős integrál számítása numerikusan, szabályos téglalap tartományon
integral3	-	Hármas integrál számítása numerikusan, szabályos téglatest tartományon
rectangle	-	Téglalap rajzolás
haltonset(n)	-	n dimenziós Halton sorozat előállítása
net(hset,n)	-	n pont kiválasztása a Halton sorozatból
inpolygon	-	egy zárt poligonon belül lévő pontok meghatározása
nnz	-	nem nulla elemek száma