Differenciálszámítás és integrálszámítás oktatása a középiskolában Maple szoftver segítségével

Szakdolgozat

Készítette: Bányász József László
V. informatika - matematika szakos hallgató

Témavezető: Dr. Gilányi Attila

Debreceni Egyetem
Matematikai Intézet
2007

Bevezetés

A differenciálszámítás és integrálszámítás rövid története

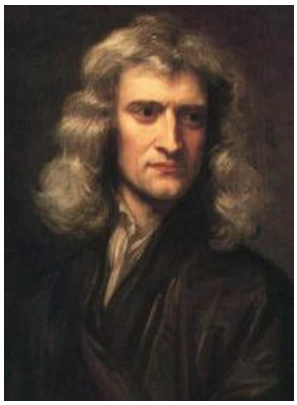
Az analízis vagy kalkulus a matematika egy jelentős területe. Módszereit arra használjuk, hogy olyan problémákat oldjunk meg, amelyeket algebrai eszközökkel nem vagyunk képesek. A kalkulus az analitikus geometriára épül. Két jelentős területe a differenciálszámítás és az integrálszámítás.

Az analízis alapjait már az ősi Görögországban és Indiában lerakták, de a modern analízis kialakulása a 17. században kezdődött Európában, *Isaac Newton* és *Gottfried Wilhelm Leibniz* munkásságának köszönhetően.

A matematika történetében *Pierre Fermat* (1601-1665) francia matematikus foglalkozott szabatosan először azzal a kérdéssel, hogy hogyan lehet egy függvény maximumát vagy minimumát meghatározni. Vizsgálatai során természetes módon jutott el a függvénygörbe érintőjéhez.

***Fermat* előkészítő munkáját követően *Gottfried Wilhelm Leibniz* (1646-1716) német matematikus az általa alapított *Acta Eruditorium* (Tudósok folyóirata) című folyóiratban 1684-ben közétette a függvénygörbék érintőjének meghatározására vonatkozó módszerét. Ezzel lényegében felfedezte a függvény differenciálhányadosát.**

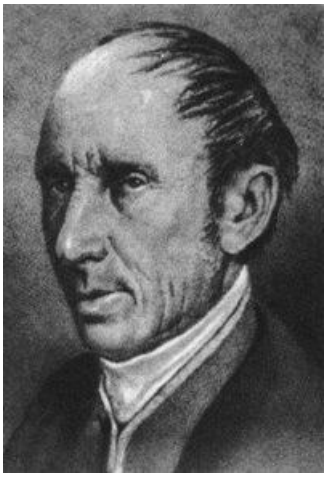
A differenciálhányados szemléletes fogalma azonban nem csak a függvénygörbe érintőjéhez kapcsolódik.



***Isaac Newton* (1643-1727) angol matematikus *Leibniztől* függetlenül, a mozgás sebességét elemezve jutott el a differenciálhányados felfedezéséhez.**

A differenciálszámítás kialakulásának korszaka így a XVI. század végétől kb. 1700-ig tartott. Szinte azt mondhatjuk, hogy eddig a korig a matematikusok egymástól többé-kevésbé függetlenül felfedezték az egyetemen jelenleg tanított kalkulus (számítási eljárás) legnagyobb részét. *Newton* és *Leibniz* mellett megemlíthetjük még *L'Hospital* francia matematikus 1696-ban kiadott *A végtelen kicsinyek analízise* című könyvét. Bár a megfogalmazásaik mai szemmel nézve pontatlanok voltak (például a határérték fogalmát a mai szabatos formában nem ismerték) és intuitív (ösztönös megérzésen alapuló) módszerrel dolgoztak, a munkásságuk mégis hallatlan nagy lökést adott a matematika fejlődésének.

A differenciálszámítás eredményes továbbfejlesztői közül talán *Euler* (1707-1783), *Bernoulli, Jacob* (1654-1705) és *Bernoulli, Johann* (1667-1748) svájci, *Taylor* (1695-1731) angol és *D'Alembert* (1716-1783) francia matematikust külön is érdemes megemlíteni. Az utóbbi sokat tett a differenciálszámítás elvi alapjainak tisztázása érdekében azzal, hogy felhívta a figyelmet a határérték fogalmára.



E fogalom körül kialakult hosszadalmas vita, bizonytalanság és pontatlanság után végre a XIX. század elején sikerült leküzdeni a nehézségeket. Ugyanis *Augustin Louis Cauchy* (1789-1857) francia matematikus 1821-ben megjelent művében közölte a határérték fogalmának ma is elfogadott szabatos definícióját, és megmutatta a szigorú alkalmazását. *Cauchynak* és követőinek köszönhető, hogy még a múlt században sikerült az analízist szilárd alapokra helyezni. Az analízis alapjainak és fogalmainak tisztázásában, a szabatos bizonyítások bevezetésében elévülhetetlen érdemei vannak *Weierstrass* (1815-1897) német matematikusnak.

Az analízis körében végzett kutatásokban különösen a XX. században értek el kiemelkedő eredményeket magyar matematikusok is. Szellemi kiválóságaink közül érdemes külön is kiemelni *Schlésinger Lajos* (1864-1933), *Reisz Frigyes* (1880-1956), *Fejér Lipót* (1880-1959) munkásságát.

Ezen matematikusok munkásságának köszönhetően az analízis a matematika komoly alapágává vált.

A Maple szoftver

A 20. század második felének meghatározó újdonsága a számítógépek megjelenése, fejlődése igazi forradalmi változást hozott számos területen. Eleinte a számítógépeket főként numerikus számolásokra használták, de már elég hamar a nagy volumenű számolásokat igénylő üzleti, menedzsment jellegű alkalmazások kezdtek dominálni. Továbbra is megmaradtak azonban a tudományos számolások, de a gépeken könnyedén, nagy gyorsasággal végrehajtható óriás mennyiségű számolást igénylő feladatok előnybe részesültek a munkaigényesebb (kézzel végzett) algebrai formula manipulációkkal szemben.

A helyzet azonban megváltozott, amikor megjelentek a "szimbolikus és algebrai" számolásra képes szoftverek, amelyek lehetővé teszik a matematikai objektumokat reprezentáló szimbólumokkal történő számolást. Ezek a szimbólumok jelölhetnek számokat (egész, racionális, valós és komplex, algebrai), de használhatók olyan matematikai objektumokra is, mint a polinomok, függvények, egyenletrendszerek, vagy más még absztraktabb matematikai struktúrák. A "szimbolikus" elnevezés azt hangsúlyozza, hogy van, amikor a választ kimondottan egy (zárt) képlet alakjában vagy egy szimbolikus közelítéssel keressük.

Az "algebrai" azt fejezi ki, hogy a számolásokat a lebegőpontos közelítő aritmetika helyett "pontosan" végezzük el, azaz pontos racionális és tetszőleges pontosságú valós aritmetikai műveletekkel. A szimbolikus és algebrai számolásra kidolgozott nyelvekre több ekvivalens elnevezés létezik: formula manipulációs, szimbolikus programozási, számítógépes algebrai stb. rendszerek. Ezek a nyelvek a szimbólumokkal végzett műveleteken, numerikus számolásokon, programozhatóságon kívül magas szintű grafikai megjelenítést is lehetővé tesznek azonban főként interaktív módon - egy vagy néhány utasítás begépelésével - használatosak. Jellemző vonásuk, hogy komoly mennyiségű és bonyolult beépített matematikai tudást tartalmaznak, matematikai szakértő rendszereknek is tekinthetők, amelyek segítségével hatékonyan és pontosan megoldható számos matematikai nyelven megfogalmazott probléma.

Előnyük, hogy leegyszerűsítik, meggyorsítják, bizonyos fokig automatizálják a matematikai probléma megoldást. Kutatási eszközként is jól használhatók. Például matematikai bizonyítások részeként: ellenőrizve vagy végrehajtva különböző bonyolult képlet átalakításokat, grafikusan ábrázolva a megoldást. A számítógépes algebrai rendszerek "matematikai kísérletek" elvégzésére is inspirálnak. Korlátaik is vannak. Például előfordul, hogy túl sok memóriát és gépidőt használnak fel, valamint a pontos aritmetika miatt gyakran exponenciálisan megnőhet egy kifejezés mérete, vagy óriási számok jelenhetnek meg. Nem veszélytelen - meglepő eredményekre is vezethet - a beépített függvények használata. Az sem mellékes, hogy értenünk kell a feladat matematikai megoldását, hogy meg tudjuk ítélni, a jó megoldást kaptuk-e meg.

Jelen dolgozatban az egyik legelterjedtebb számítógépes algebrai nyelv - a Maple - gazdag lehetőségeit alkalmazzuk. A dolgozat írásakor a kanadai Waterloo Maple Software Co. már a Maple 11 verziónál tart.

Mi is az a Maple?

A Maple egy magas szintű komputer algebrai és vizuális megjelenítésre alkalmas rendszer. Tökéletes környezetet biztosít szimbolikus formulák szimbolikus átalakításához, algebrai kifejezésekkel való operáláshoz, gyakorlatilag tetszőleges pontosságú számoláshoz, és a legkülönbözőbb két és háromdimenziós ábrák elkészítéséhez. Egyik fő ereje hogy a rendszer lehetőségeit és "tudását" szinte korlátlanul lehet bővíteni. Így széles körben alkalmazható a matematika legkülönbözőbb ágaiban, az oktatásban, ezen kívül a mérnöki, üzleti és gazdasági életben egyaránt.

A Maple-lel egy munkalapon keresztül lehet kommunikálni, az utasításokat a munkalap aktuális helyén található parancssorból tudjuk kiadni. Az aktuális parancssor alá írja ki a válaszait (számítási eredmények, hibaüzenetek, stb.), és egy külön ablakba kerülnek az ábrák, animációk. Ezeket persze bemásolhatjuk a munkalapra, ott lehet őket szerkeszteni a táblázatokkal, számítási eredményekkel és sok minden mással együtt, és tetszetős formában ki lehet őket nyomtatni, vagy más célra felhasználni.

Megjegyzés a nyomtatott verzióhoz:

Szakdolgozatom számítógépes változatában teljes értékű, ugyanis a Maple szoftver és a HTML nyelv lehetőségeit kihasználva, sok animációt és néhány felhasználóval kommunikáló, interaktív eljárást tartalmaz, amelyeket nyomtatott formában sajnálatos módon, nem tud visszaadni a dolgozat. Természetesen az animációk és az inetraktív eljáráshívások is bekerültek a nyomtatott verzióba és ezek külön jelölve vannak. Az animációknál a legjellemzőbb képkocka (általában az utolsó) szerepel állóképként, eljárásoknál pedig egy kommunikációs példa. Ezek fényében javasolni tudom, a dolgozatom HTML verziójának számítógépen való megtekintését, mivel az tekinthető a valós, teljes értékű dolgozatnak.

Bevezetés a differenciálszámításba

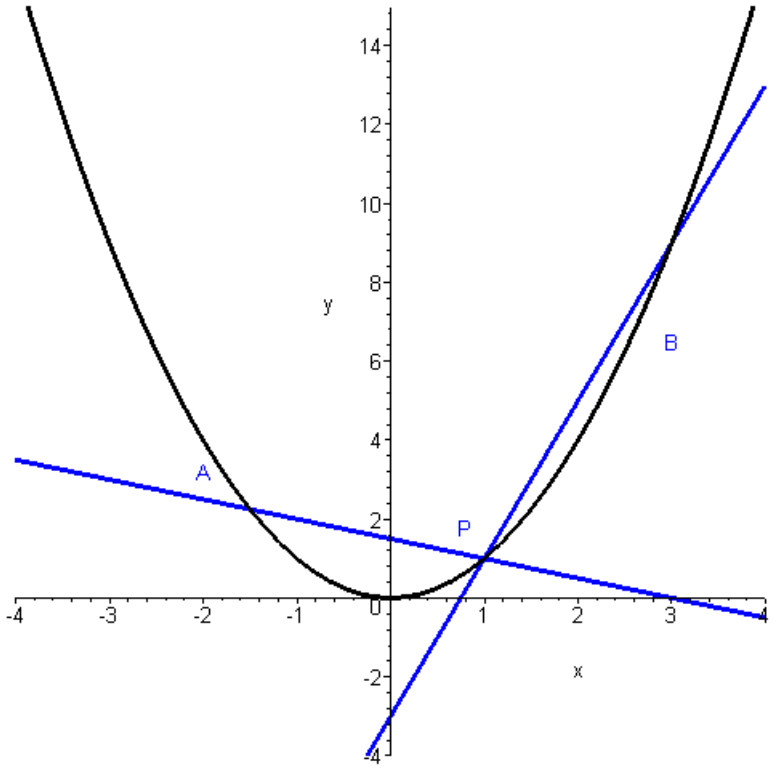
Az érintő szemléletes fogalma

Lineáris függvény esetén:

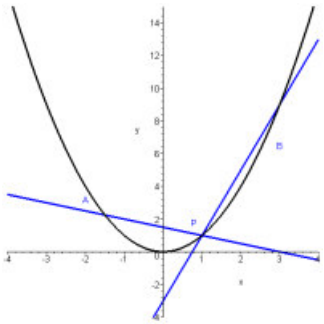
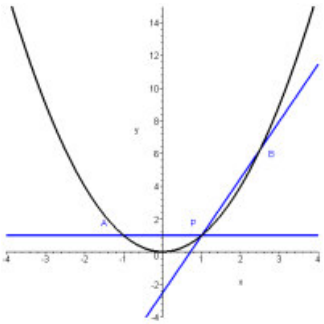
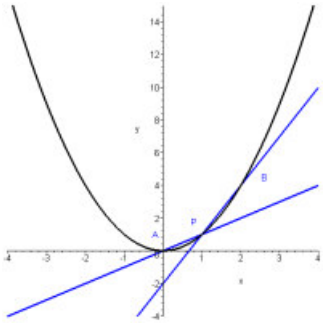
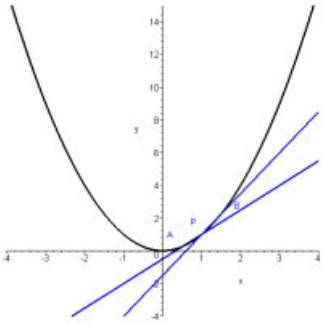
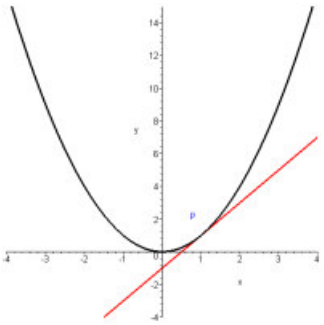
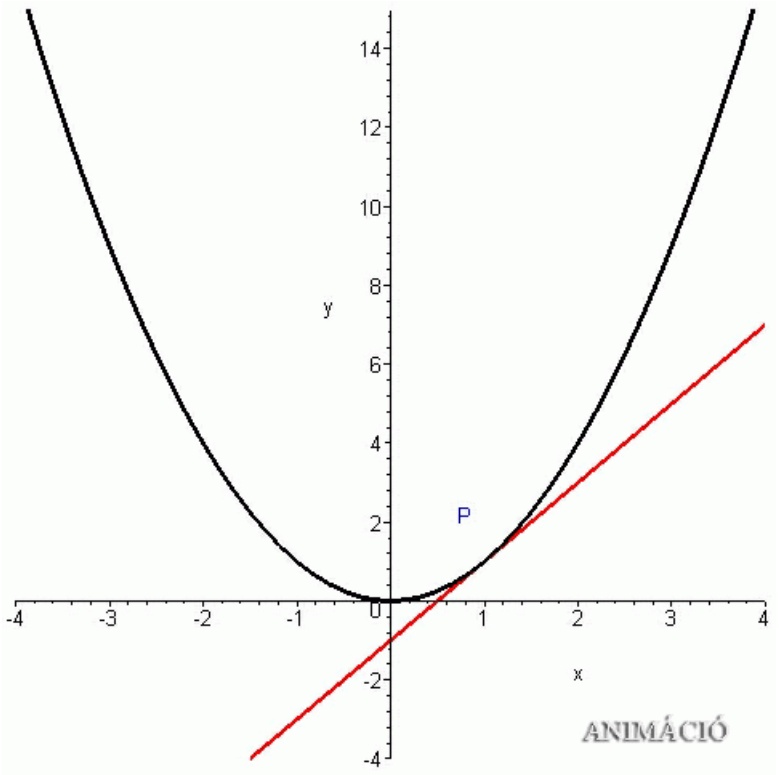
Lineáris függvény képe egyenes. Ismereteink alapján, az egyenes, a függvénygörbe irányát, adott P pontban a derékszögű háromszög befogóinak a hányadosával, az egyenes irányszögének a tangensével jellemezhetjük (iránytangens).

Másodfokú függvény esetén:

Szeretnénk pontosan követni a függvénygörbe irányát, a függvény menetét nem lineáris függvények esetén is. Tekintsük a parabolát, és értelmezzük a parabola irányát adott P pontban. Kössük össze a P pontot a parabola egy tőle balra eső A pontjával és a parabola egy tőle jobbra eső B pontjával. Sem az AP egyenes, sem a BP egyenes irányát nem fogadhatjuk el a parabola irányának a P pontban. Mert ha elfogadnánk őket, akkor egyrészt a parabolának a P pontban két iránya lenne, másrészt a két irány változna , ha mind az A pontot, mind a B pontot közelítenénk a P ponthoz.

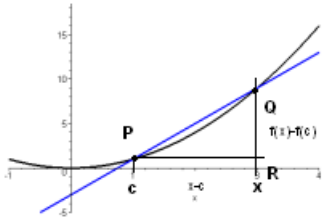


De a B vagy az A pontnak a P pont felé történő mozgásakor lesz olyan pillanat, amikor a B, illetve az A pont a P pontba érkezik. Ekkor a BP szelő, illetve az AP szelő "elpattan" a parabolától. A szemlélet az sugallja, hogy a két elpattanó szelő egybeesik, azonos lesz. A szelőkből egyetlen érintő lesz a P pontban, célszerű ezzel az érintővel jellemezni a parabola irányát.







y
x
P
ANIMÁCIÓ

Az érintő definiálása

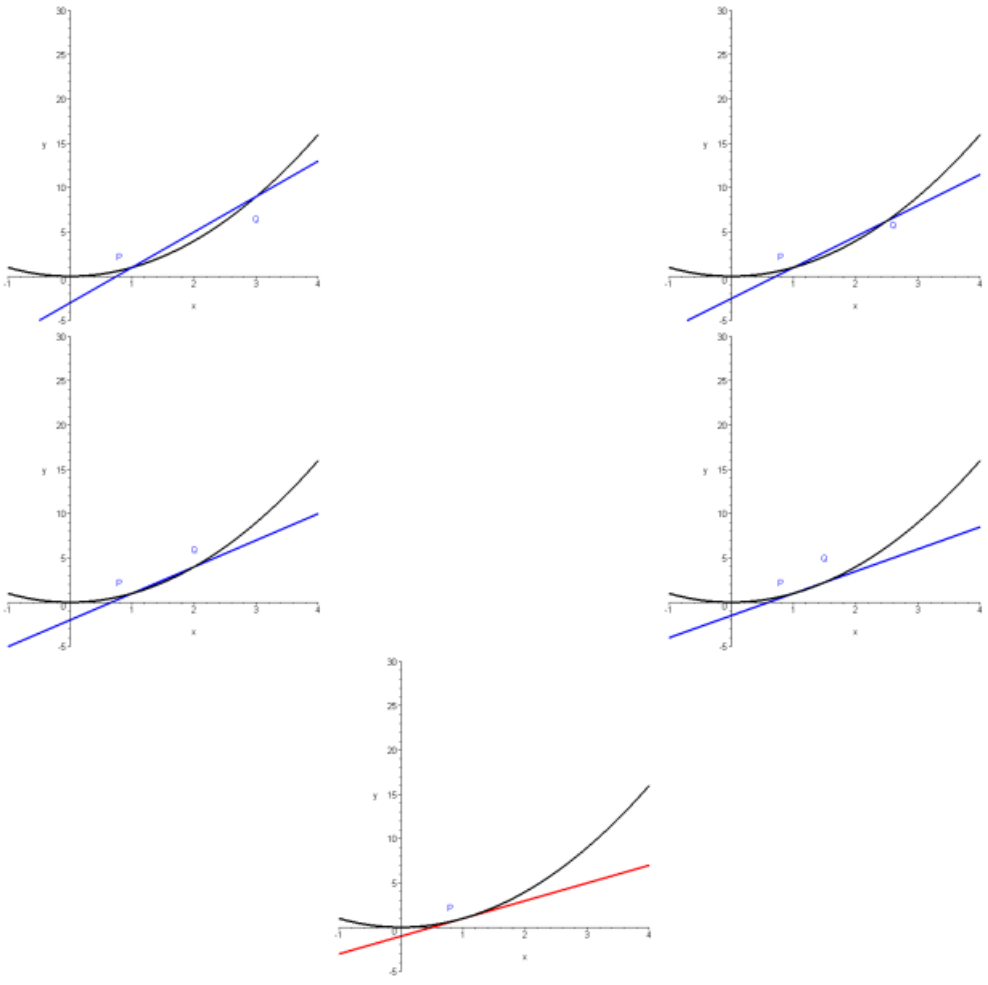
Legyen S adott intervallum. Az f: S → R függvény grafikonjának két pontját jelölje a P(c, f (c)) és Q(x, f (x)), ahol $c \in S$.

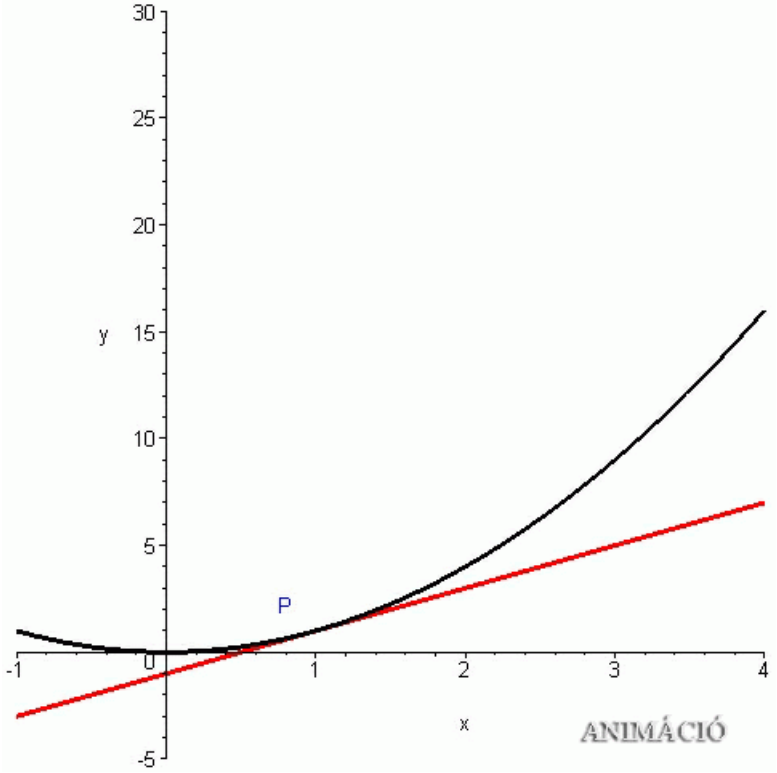


A PQ szelő meredekségét a PQ és az Ox tengely pozitív iránya által bezárt szög tangense adja, s mivel PR párhuzamos az x-tengellyel, ezért ez a szög a QPR szöggel egyenlő, tehát

$$tgQPR\angle = \frac{QR}{PR} = \frac{f(x) - f(c)}{x - c}$$

Ha ez a differenciahányados egy m határértékhez tart, miközben $x \in S$ tart c-hez, akkor a szelő a P pontra illeszkedő m meredekségű egyeneshez tart, melyet az y = f(x) egyenletű görbe c abszcisszájú pontjához tartozó érintőjének nevezzük.





A geometriai szemlélettől függetleníthetjük azt az eljárást, amely szerint az f és c ismeretében meghatározzuk az m számértéket.

Az f: S → R függvénynek differenciálhányadosa a $c \in S$ helyen m (akkor és csak akkor), ha

$$\lim_{x \to c} \frac{f(x) - f(c)}{x - c} = m$$

Ekkor röviden azt mondjuk, hogy f differenciálható a c pontban.

Függvények és érintőik

A következő függvényeket tekinthetjük meg:

Alapfüggvények:

- **Másodfokú függvény**
- **Négyzetgyökfüggvény**
- **Exponenciális függvény**
- **Logaritmusfüggvény**
- **Szinusz függvény**
- **Koszinusz függvény**

Továbbá néhány, érdekes függvény

A függvények és érintőik ábrázolását az alábbi, saját magam készítette eljárások segítségével teszem:

```
> restart: with(plots):
```

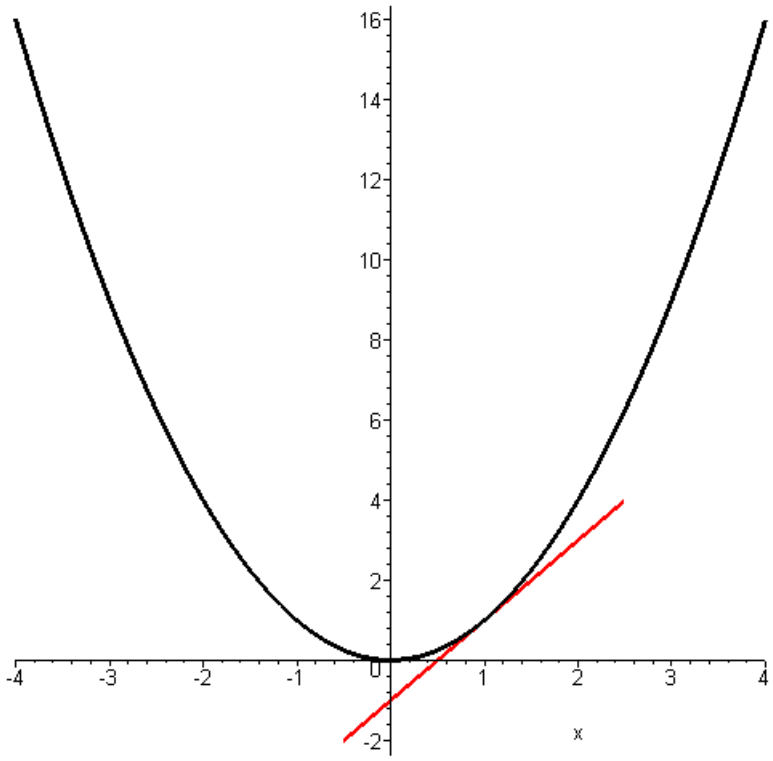
```
> rajzol := proc(f,k,v,p)
  local fv, erinto, df, p1, p2,kezdo,veg,po:
  fv:=f;
  kezdo:=k;
  veg:=v;
  po:=p;
  df:=diff(fv,x);
  p1:=plot(fv, x=kezdo..veg, scaling=unconstrained,thickness=3, color=black):
  erinto:=subs(x=a, df)*(x-a)+subs(x=a,fv);
  p2:=plot(subs(a=po, erinto), x=(po-1.5)..(po+1.5), color=red, thickness=3):
  display({p1, p2});
  end proc:
```

Ez az eljárás a függvény egy adott pontjában meghúzza az érintőjét.

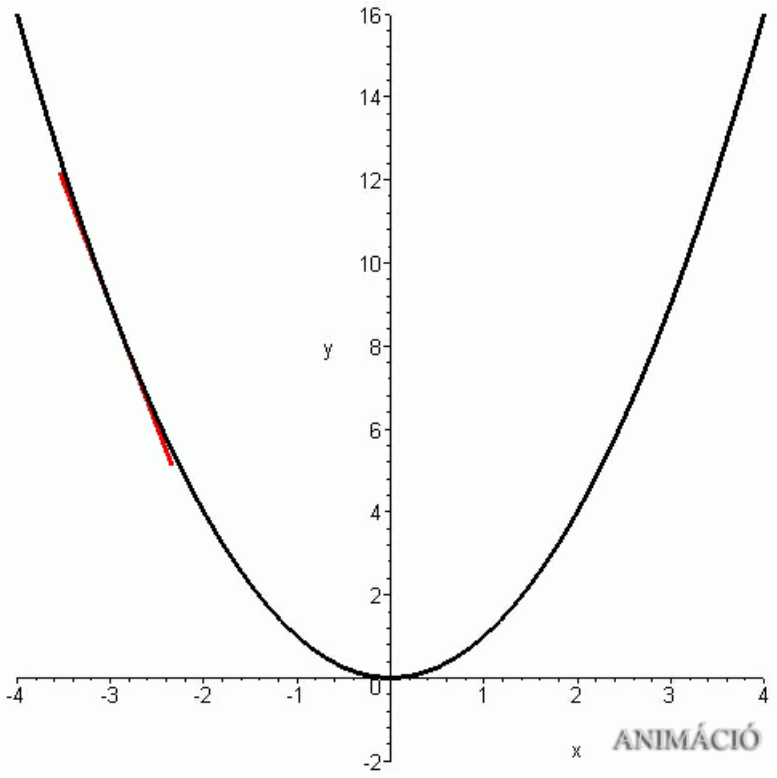
```
> rajzol2 := proc(f,kx,vx,ky,vy,e)
  local fv, erinto, df, p1, p2,kezdox,vegx,kezdoy,vegy,esz,i:
  fv:=f;
  kezdox:=kx;
  vegx:=vx;
  kezdoy:=ky;
  vegy:=vy;
  esz:=e;
  df:=diff(fv,x);
  p1:=plot(fv, x=kezdox..vegx,y=kezdoy..vegy, scaling=unconstrained,thickness=3, color=black):
  erinto:=subs(x=a, df)*(x-a)+subs(x=a,f);
  koz:=(vegx-kezdox)/esz;
  for i from 1 to esz do
  lista[i]:=display(p1,plot(subs(a=-3+i*6/esz, erinto),x=-3+i*6/esz-0.6..-3+i*6/esz+0.6,y=kezdoy..vegy, color=red)):
  od:
  display(seq(lista[i], i=1..esz), insequence=true,thickness=3, scaling=unconstrained);
  end proc:
```

Ez az eljárás meghúzza a függvény adott számú az érintőjét, és animációban ábrázolja.

> **rajzol(x^2,-4,4,1);**

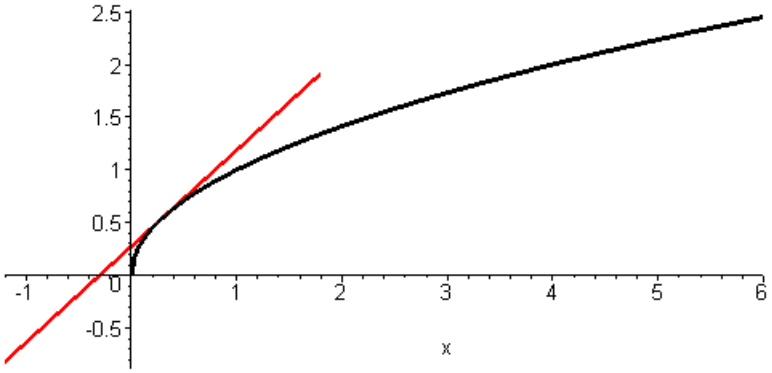


> **rajzol2(x^2,-4,4,-2,16,100);**

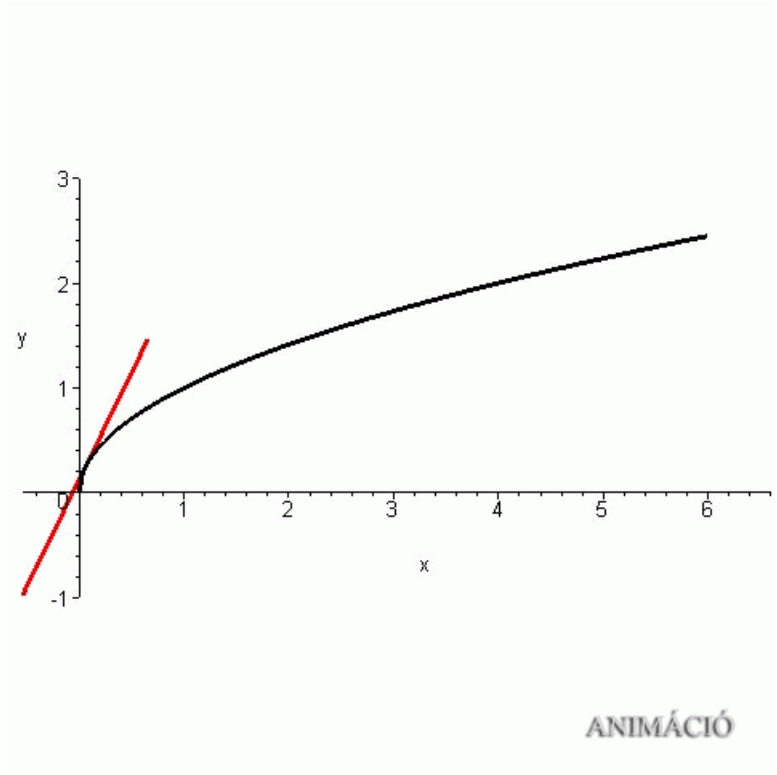


>

> **rajzol(sqrt(x),0,6,0.3);**

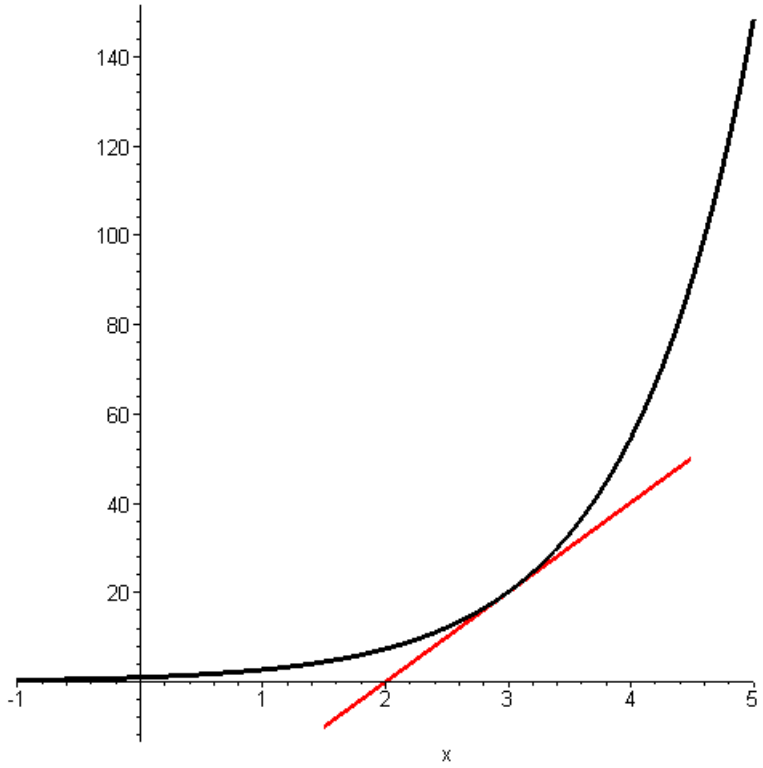


> **rajzol2(sqrt(x),0,6,-1,3,100);**

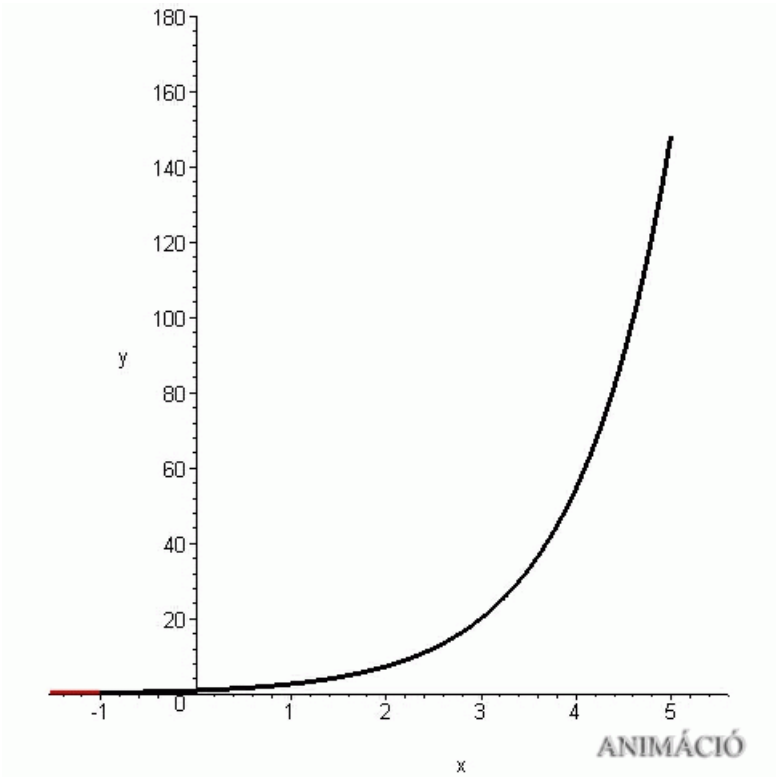


>

> **rajzol(exp(x),-1,5,3);**

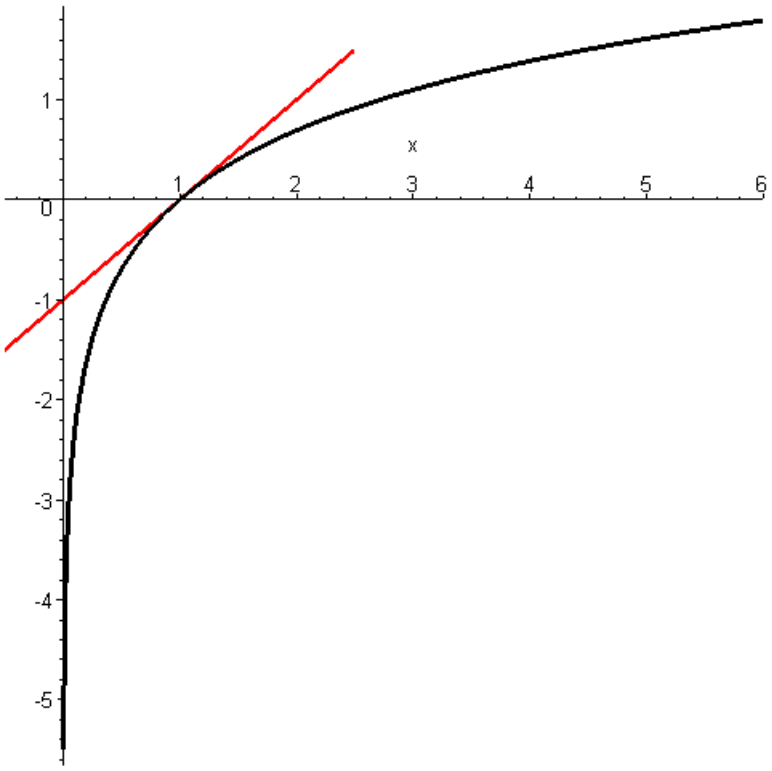


> **rajzol2(exp(x),-1,5,0,180,100);**

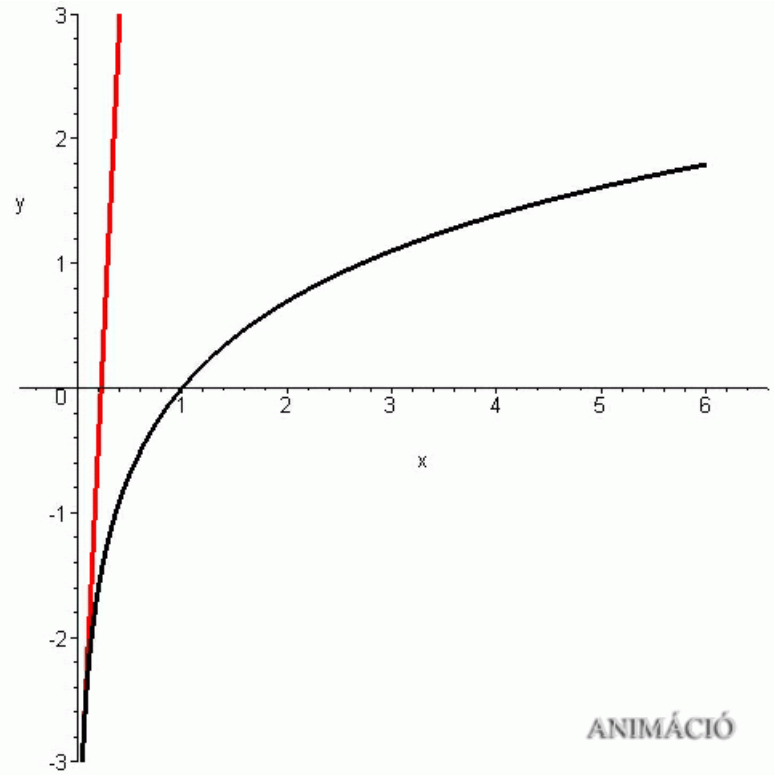


>

> **rajzol(log(x),0,6,1);**

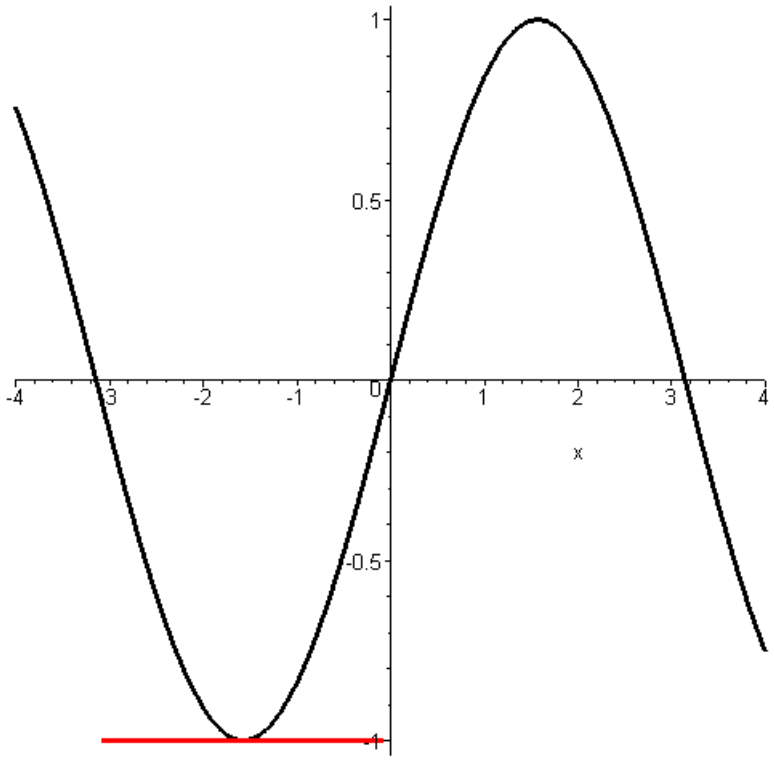


> **rajzol2(log(x),0,6,-3,3,100);**

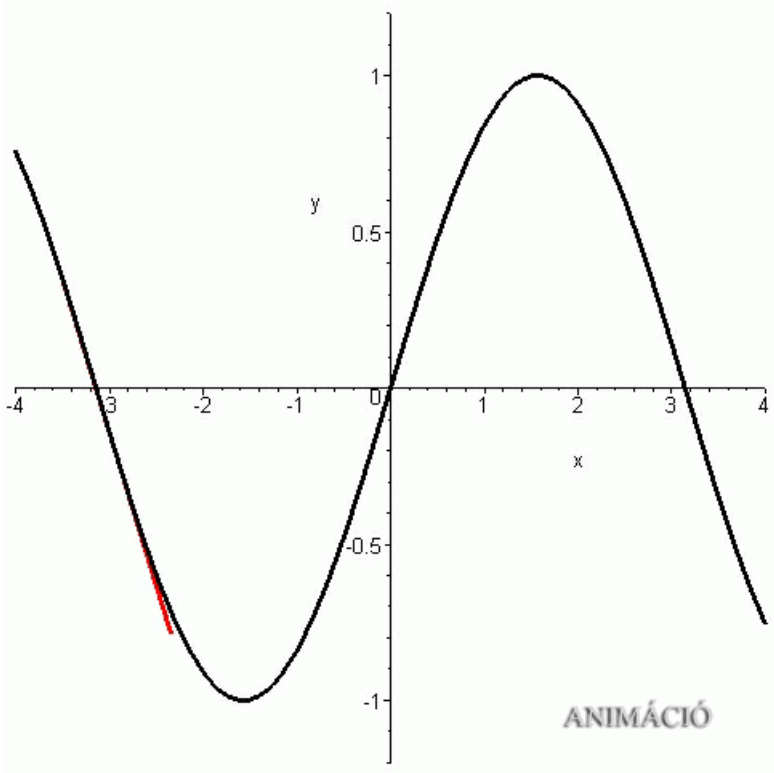


>

> **rajzol(sin(x),-4,4,-Pi/2);**

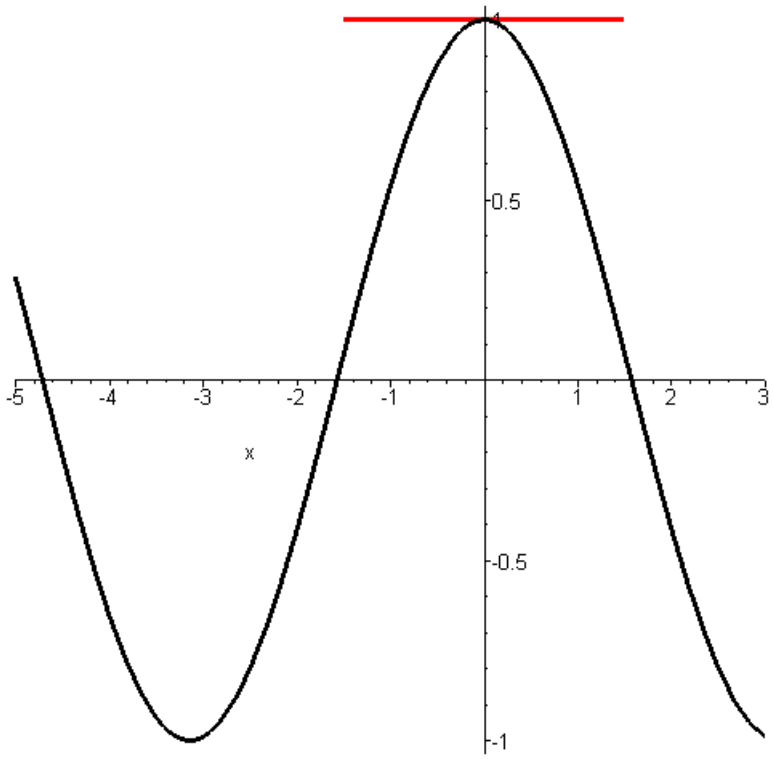


> **rajzol2(sin(x),-4,4,-1.2,1.2,100);**

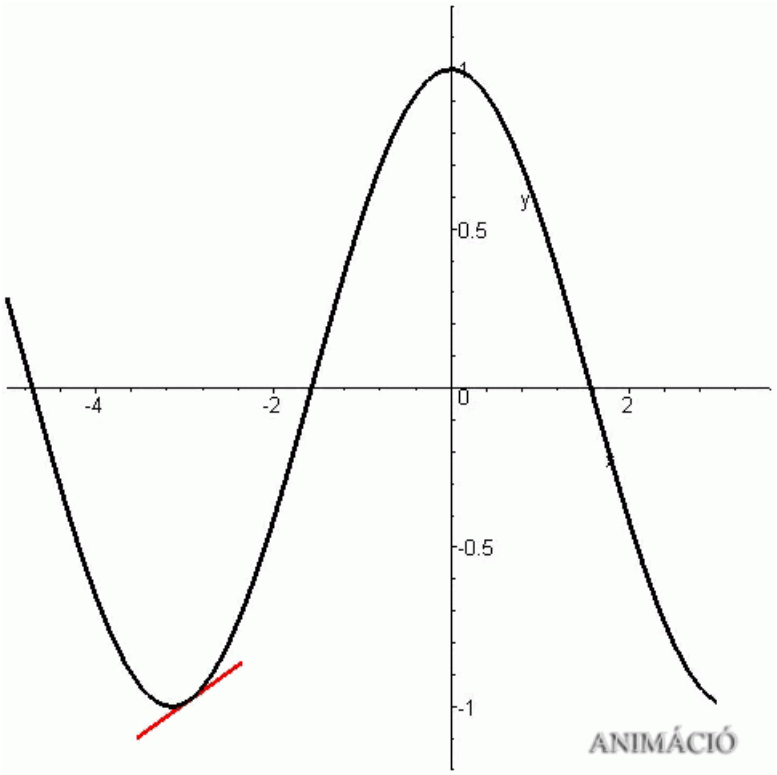


>

> **rajzol(cos(x),-5,3,0);**

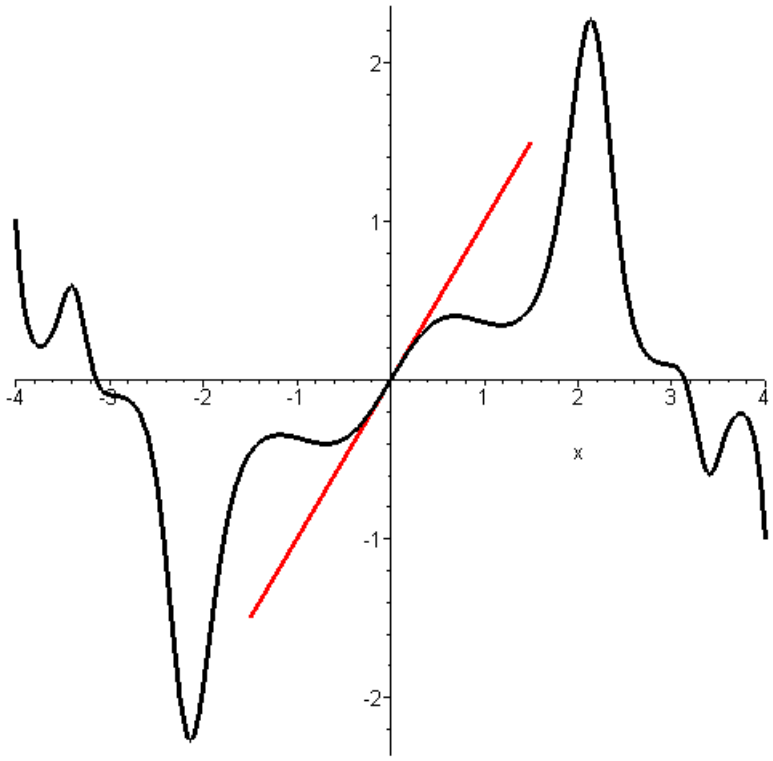


> **rajzol2(cos(x),-5,3,-1.2,1.2,100);**

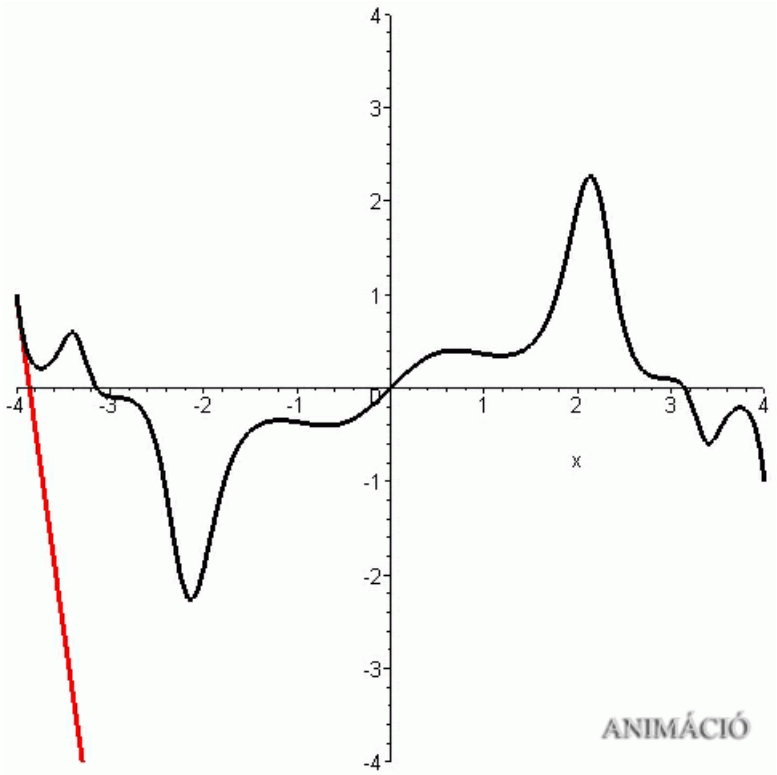


>

> **rajzol(exp(-sin(x^2))*sin(x),-4,4,0);**

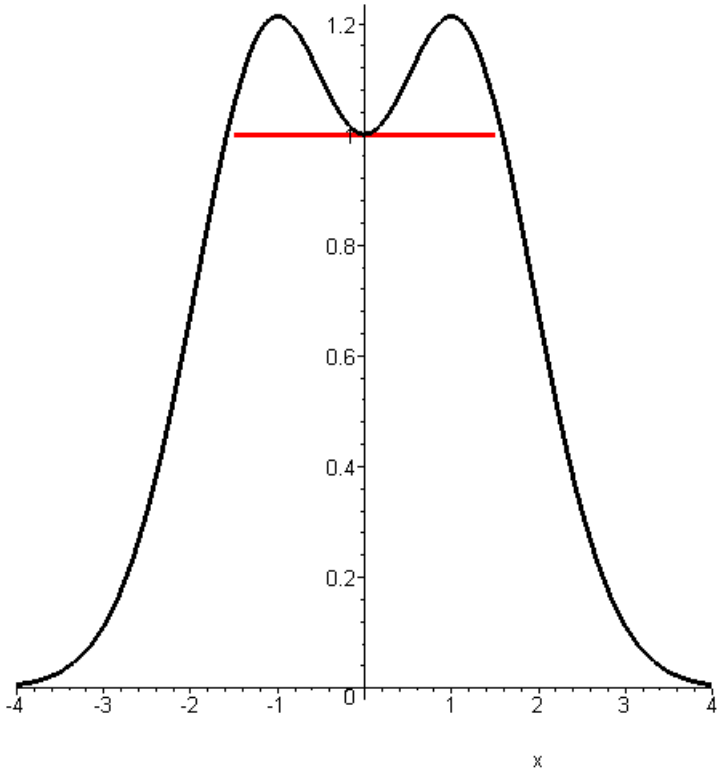


> **rajzol2(exp(-sin(x^2))*sin(x),-4,4,-2.5,2.5,100);**

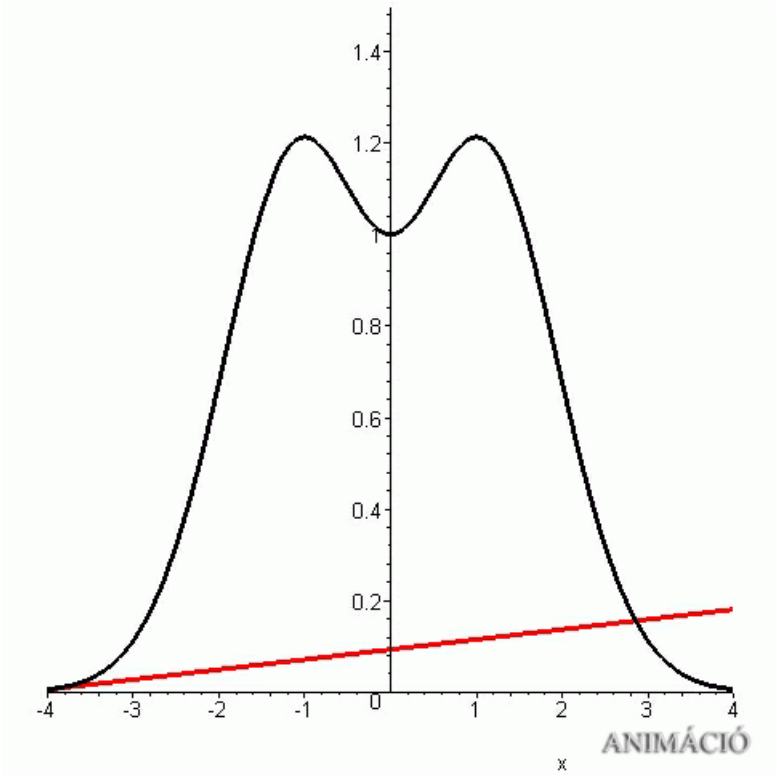


>

> **rajzol((1+x^2)*exp(-x^2/2),-4,4,0);**

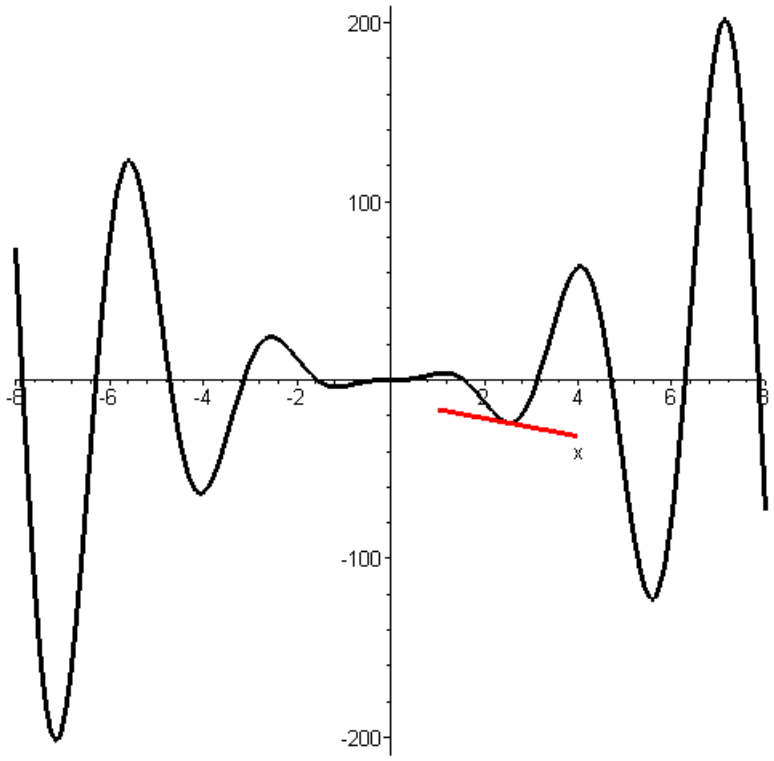


> **rajzol2((1+x^2)*exp(-x^2/2),-4,4,0,1.3,100);**

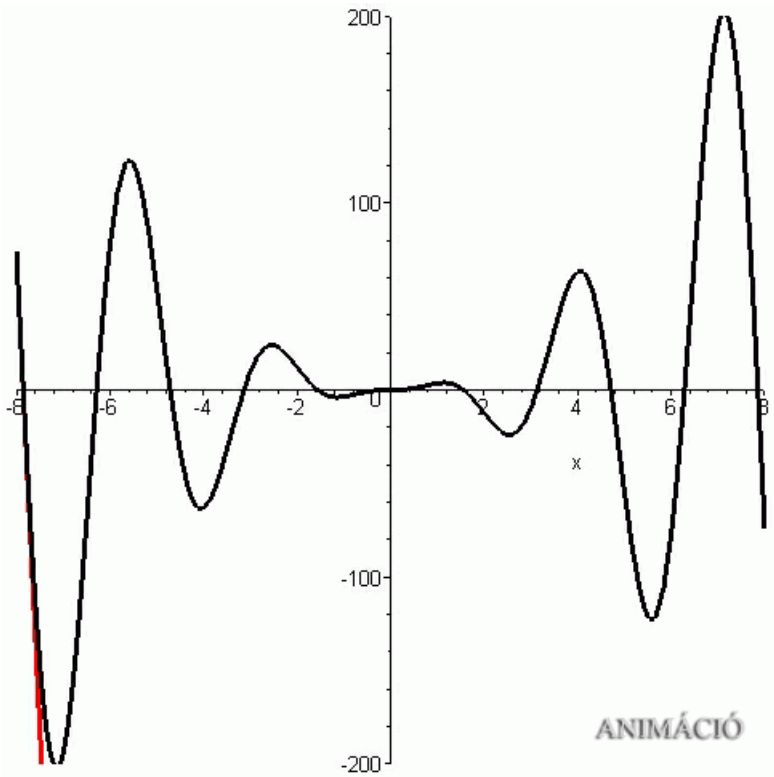


>

> **rajzol(sin(2*x)*(4*x^2),-8,8,2.5);**

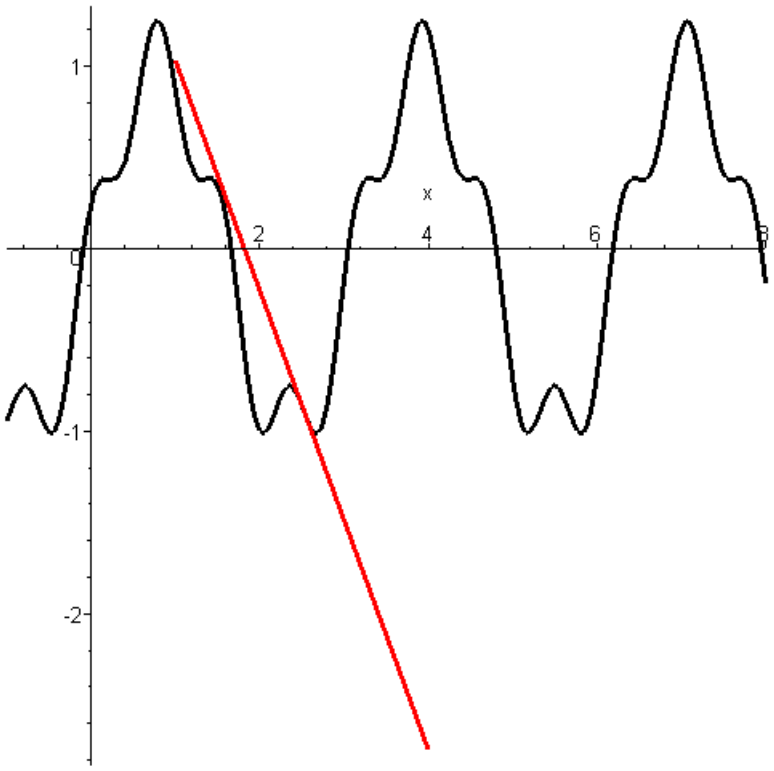


> **rajzol2(sin(2*x)*(4*x^2),-8,8,-210,210,100);**

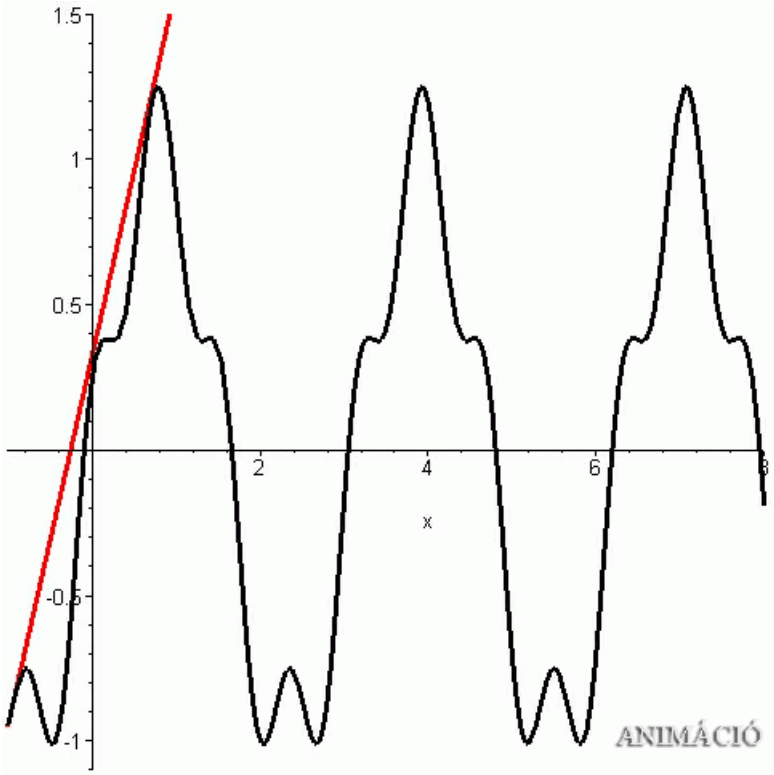


>

> **rajzol(sin(2*x)+0.25*cos(8*x),-1,8,2.5);**



> **rajzol2(sin(2*x)+0.25*cos(8*x),-1,8,-1.1,1.5,100);**



>

Egy Ön által választott függvény

Interaktív rész

Próbálja ki a Maple szoftvert!

Előre megírt eljárások segítségével ábrázolhat egy Ön által megadott függvényt és meghúzhatja egy adott pontban az érintőjét, vagy adott számú érintőt húzhat a függvényhez.

Az eljárás futtatásához telepített Maple szoftverre van szükség!

- **Egy érintő egy adott pontban**
- **Érintők egy adott függvényhez**
- **Érintők egy adott függvényhez x és y tengely ábrázolási intervallum megadásával**

A következő oldalakon az eljárások forráskódja, egy kommunikációs példa és annak eredménye látható.

Egy érintő egy adott pontban

Egy Ön által megadott függvényhez egy adott pontban érintőt húz az eljárás.

A promptok után az Enter billentyű leütésével kell futtatnia az utasításokat (sorban mind a hármat).

A rajzol( ) ; után adja meg a kiválasztott függvény adatait.

FONTOS: az adatok bevitelét ; -el zárja le és csak azután üsse le az Enter billentyűt.

Függvény megadása pl: sin(x); x^2+3; exp(x);

```
> restart: with(plots):
```

```
> rajzol := proc()
  local fv, erinto, df, p1, p2,kezdo,veg,po:
  fv:=readstat("Melyik függvényt ábrázoljuk: ");
  kezdo:=readstat("A függvényt mely intervallumban ábrázoljuk? A kezdőpontja:");
  veg:=readstat("Végpontja: ");
  po:=readstat("Melyik pontban húzzuk meg az érintőt: ");

  df:=diff(fv,x);
  p1:=plot(fv, x=kezdo..veg, scaling=unconstrained,thickness=3, color=black):
  erinto:=subs(x=a, df)*(x-a)+subs(x=a,fv);
  p2:=plot(subs(a=po, erinto), x=(po-1.5)..(po+1.5), color=red, thickness=3):
  display({p1, p2});

  end proc:
```

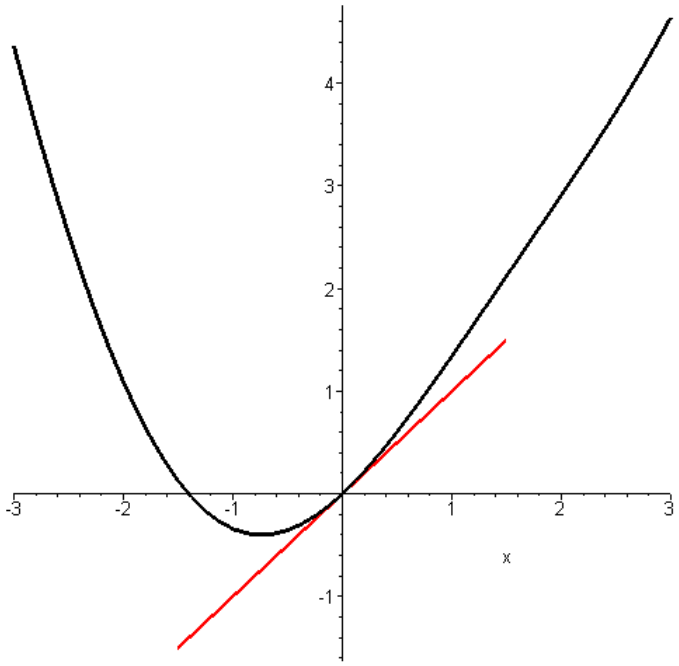
```
> rajzol();
```

Melyik függvényt ábrázoljuk: **x^2/2+sin(x);**

A függvényt mely intervallumban ábrázoljuk? A kezdőpontja: **-3;**

Végpontja: **3;**

Melyik pontban húzzuk meg az érintőt: **0;**



>

Érintők adott függvényhez

Egy Ön által megadott függvényhez megadott számú érintőt húz az eljárás.

A promptok után az Enter billentyű leütésével kell futtatnia az utasításokat (sorban mind a hármat).

A rajzol2( ) ; után adja meg a kiválasztott függvény adatait.

FONTOS: az adatok bevitelét ; -el zárja le és csak azután üsse le az Enter billentyűt.

Függvény megadása pl: sin(x); x^2+3; exp(x);

Ha a jobban láthatóság miatt, meg szeretné adni, hogy az eljárás az abszcissza- (x) és a ordináta (y) tengelyek mely részét ábrázolja akkor az Érintők egy adott függvényhez y tengely ábrázolási intervallum megadásával - munkalapon a rajzol3 eljárást hívja meg.

```
> restart:with(plots):
```

```
Warning, the name changecoords has been redefined
```

```
> rajzol2 := proc()
  local f, erinto, df, p1, p2,kezdo,veg,esz,i:
  f:=readstat("Melyik függvényt ábrázoljuk: ");
  kezdo:=readstat("A függvényt mely intervallumban ábrázoljuk? A kezdőpontja:");
  veg:=readstat("Végpontja: ");
  esz:=readstat("Hány érintőt húzzunk meg: ");

  df:=diff(f,x);
  p1:=plot(f, x=kezdo..veg, scaling=unconstrained,thickness=3, color=black):
  erinto:=subs(x=a, df)*(x-a)+subs(x=a,f);
  koz:=(veg-kezdo)/esz;

  for i from 1 to esz do
  lista[i]:=display(p1,plot(subs(a=kezdo+i*koz, erinto), x=kezdo..veg, color=red)):
  od:

  display(seq(lista[i], i=1..esz), insequence=true,thickness=3, scaling=unconstrained);
  end proc:
```

```
Warning, `koz` is implicitly declared local to procedure `rajzol2`
```

```
Warning, `lista` is implicitly declared local to procedure `rajzol2`
```

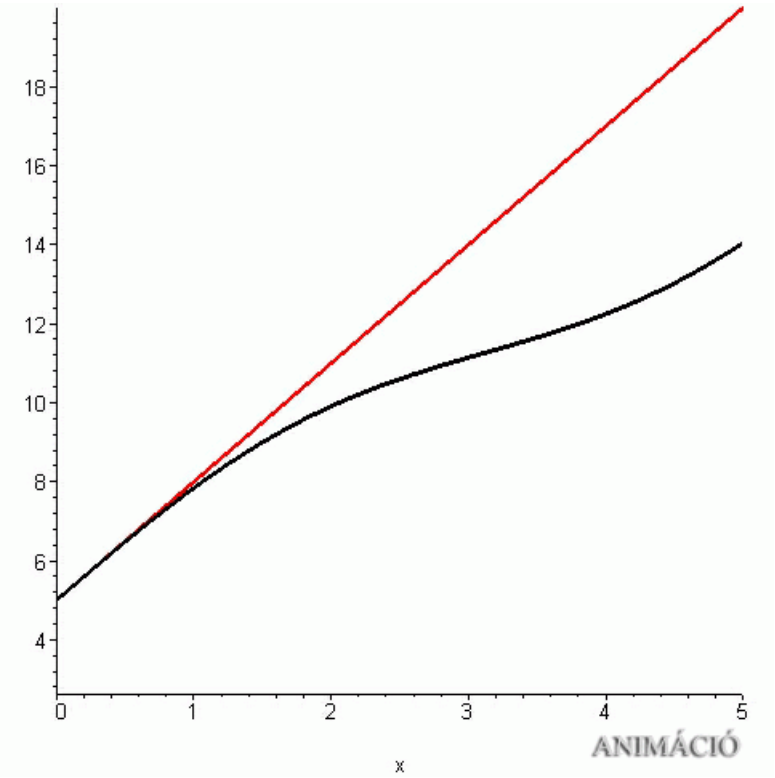
```
> rajzol2();
```

Melyik függvényt ábrázoljuk: **x*2+sin(x)+5;**

A függvényt mely intervallumban ábrázoljuk? A kezdőpontja: **0;**

Végpontja: **5;**

Hány érintőt húzzunk meg: **100;**


18
16
14
12
10
8
6
4
0
1
2
3
4
5
x
ANIMÁCIÓ

>

Érintők adott függvényhez x és y tengely ábrázolási intervallum megadásával

Egy Ön által megadott függvényhez megadott számú érintőt húz az eljárás.

Megadhatja, hogy az eljárás az abszcissza- (x) és a ordináta (y) tengelyek mely részét ábrázolja.

A promptok után az Enter billentyű leütésével kell futtatnia az utasításokat (sorban mind a hármat).

A rajzol3( ) ; után adja meg a kiválasztott függvény adatait.

FONTOS: az adatok bevitelét ; -el zárja le és csak azután üsse le az Enter billentyűt.

Függvény megadása pl: sin(x); x^2+3; exp(x);

```
> restart:with(plots):

Warning, the name changecoords has been redefined


> rajzol3 := proc()
  local f, erinto, df, p1, p2,kezdox,vegx,kezdoy,vegy,esz,i:
  f:=readstat("Melyik függvényt ábrázoljuk: ");
  kezdox:=readstat("A függvényt mely intervallumban ábrázoljuk? Az x tengely kezdőpontja:");
  vegx:=readstat("Az x tengely végpontja: ");
  kezdoy:=readstat("Az y tengely kezdőpontja: ");
  vegy:=readstat("Az y tengely végpontja: ");
  esz:=readstat("Hány érintőt húzzunk meg: ");

  df:=diff(f,x);
  p1:=plot(f, x=kezdox..vegx,y=kezdoy..vegy, scaling=unconstrained,thickness=3, color=black):
  erinto:=subs(x=a, df)*(x-a)+subs(x=a,f);
  koz:=(vegx-kezdox)/esz;

  for i from 1 to esz do
  lista[i]:=display(p1,plot(subs(a=kezdox+i*koz, erinto), x=kezdox..vegx,y=kezdoy..vegy, color=red)):
  od:

  display(seq(lista[i], i=1..esz), insequence=true,thickness=3, scaling=unconstrained);
  end proc:

Warning, `koz` is implicitly declared local to procedure `rajzol3`


Warning, `lista` is implicitly declared local to procedure `rajzol3`


> rajzol3();
```

Melyik függvényt ábrázoljuk: **cos(x)/2;**

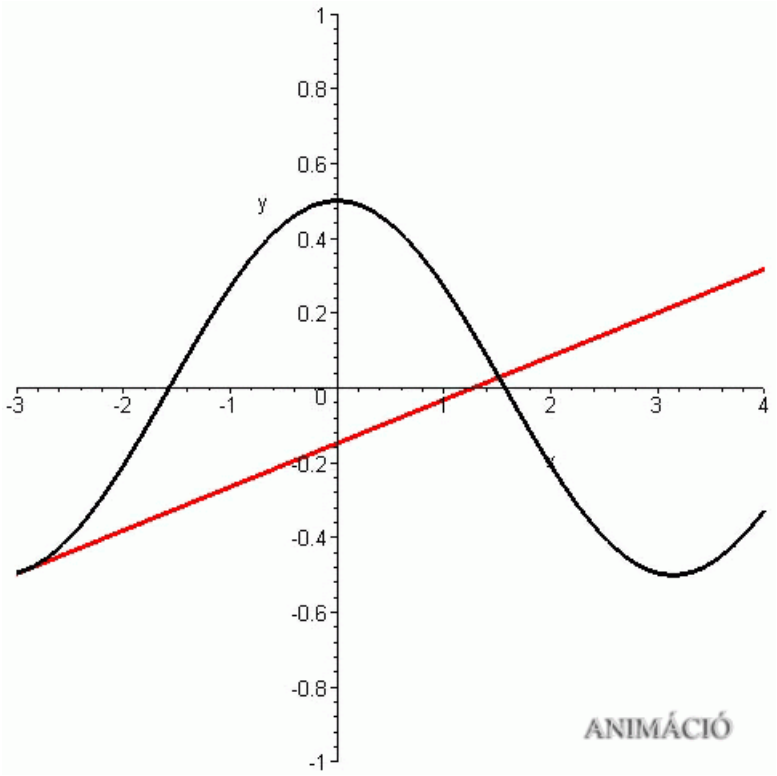
A függvényt mely intervallumban ábrázoljuk? Az x tengely kezdőpontja: **-3;**

Az x tengely végpontja: **4;**

Az y tengely kezdőpontja: **-1;**

Az y tengely végpontja: **1;**

Hány érintőt húzzunk meg: **75;**


1
0.8
0.6
y
0.4
0.2
-3
-2
-1
0
1
2
3
4
x
-0.2
-0.4
-0.6
-0.8
-1
ANIMÁCIÓ

>

A differenciálszámítás a felsőbb matematikában

A fejezet eddigi része a differenciálszámítás bevezetéséről szólt, a középiskolai tananyagra és tárgyalás módszereire támaszkodva.

Aki szeretné megtekinteni, tanulmányozni a differenciálszámítás magasabb szinten való tárgyalását az Dr. Lajkó Károly, egyetemi docens Analízis 2 jegyzetéből megteheti.

A jegyzet az alábbi linken érhető el: http://www.math.klte.hu/~lajko/

Bevezetés az integrálszámításba

Kétoldali közelítés

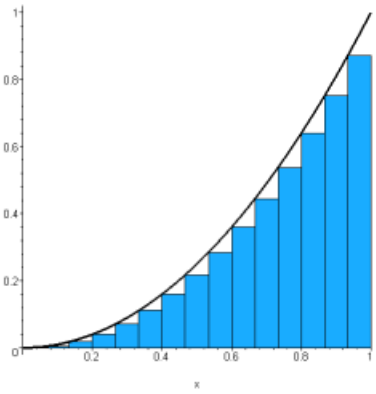
Síkidomok esetében:

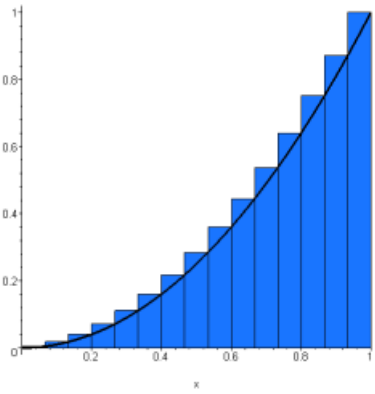
A kör területét a beírt és körülírt n (≥ 4) oldalú szabályos sokszögek területeivel közelítettük meg. A kör területének kiszámítása után felvethető az a kérdés, hogy hogyan lehet kiszámolni olyan síkidom területét, amelyet tetszőleges görbe vonal határol. Hasznos gondolatnak látszik, ha a síkidomot egyenes szakaszokkal szétdaraboljuk, azután kiszámítjuk a részidomok területét (a részidomokat három vagy két oldalról egyenes szakaszok és egyik oldalról görbe vonal határolják), majd a kapott területeket összeadjuk. De hogyan számoljuk ki egy részidom területét? Mivel egy görbe vonal biztosan van a részidomunkban, ezért a területének kiszámítása nem egyszerű dolog.
A kör területét beírható és körülírható szabályos sokszögek területével közelítettük meg. Tetszőleges síkidomnál a görbe vonalak miatt szabályos sokszögekkel nem számolhatunk, de eljárhatunk úgy, hogy a síkidomot felparcellázzuk és az egyes parcellákba téglalapokat illesztünk, illetve ez egyes parcellákat tartalmazó téglalapokat jelölünk ki. Nyilvánvaló, hogy ekkor a síkidom területe a síkidom által tartalmazott téglalapok területének összege és a síkidomot tartalmazó téglalapok területének összege közé esik.
A téglalapok területének kiszámításához két adatot kell ismernünk: az alapot és a magasságot. A görbe vonalú síkidom területét nagyobb pontossággal határozhatjuk meg ha a parcellák számát szaporítjuk.

Függvények esetében:

Most számítsuk ki annak az OAB zárt síkidomnak a területét, amelyet felülről az $y=x^2$ egyenletű parabola íve, alulról az x tengely, oldalról az x tengelyre merőleges AB szakasz (ordináta) határol. A síkidomot parabolikus háromszögnek nevezzük.
Feladatunkat átfogalmazhatjuk a következőképpen:
Számítsuk ki az f: R→R, $f(x)=x^2$ függvény görbéje alatti területet a [0;1] intervallumban. Vagy azt is mondhatjuk, hogy számítsuk ki az f: R→R, $f(x)=x^2$ függvény görbéje alatti területet az $x_1=0$ és az $x_2=1$ határok között.
Osszuk fel a [0;1] intervallumot, az OA szakaszt 4 egyenlő részre.

- **Emeljünk a szomszédos osztópontok által meghatározott szakaszok fölé téglalapokat, olyan módon, hogy a téglalap magassága az adott szakaszon a függvény legkisebb értékével legyen egyenlő. Ha a 4 téglalap területét összeadjuk akkor megkapjuk a beírt téglalapok területének összegét.**
- **Aztán a szakaszok fölé emeljünk téglalapokat olyan módon, hogy az adott szakaszon a téglalap magassága most a legnagyobb függvényérték legyen. Ha ezeket is összeadjuk akkor a körülírt téglalapok területének összegét kapjuk meg.**





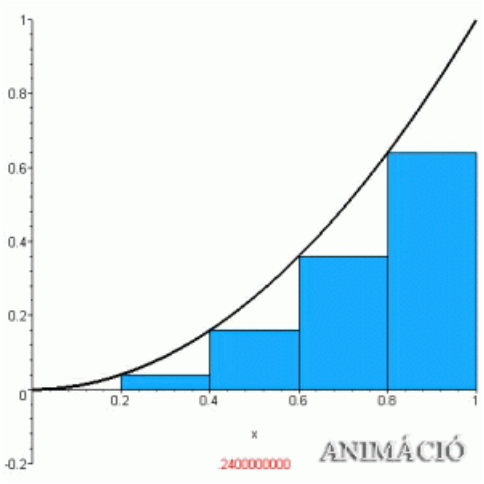
A beírt téglalapok területének összegéből hiányzik valamennyi, hogy a teljes területet kapjuk, míg a körülírt téglalapok területösszege valamennyivel több, mint a függvény alatti terület. A szemlélet azt sugallja, hogy a hiányt és a többletet is csökkenthetjük ha a [0;1] intervallum felosztását növeljük. Ezt az eljárást a matematikában szaknyelven a felosztás finomításának nevezzük.

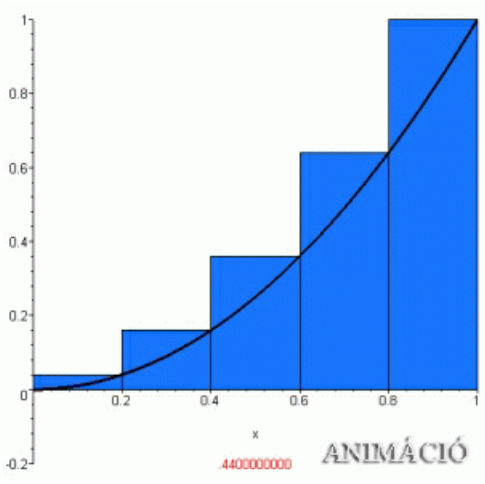
Általánosítsunk:

Osszuk fel a a [0;1] intervallumot n egyenlő részre és számítsuk ki a beírt, ill. a körülírt téglalapok területeinek összegét. A beírt téglalapok területeinek összegét jelöljük s_n-el, a körülírt téglalapok területeinek összegét jelöljük S_n-el. (s illetve S a latin "summa", magyarul "összeg" szó kezdőbetűje. Az n index pedig arra utal, hogy a [0;1] intervallumot n egyenlő részre bontottuk).

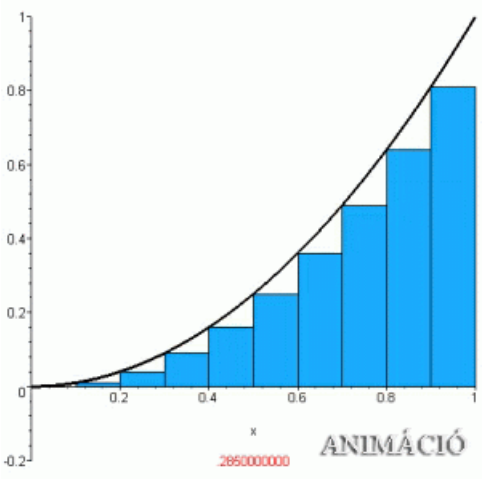
Példák: különböző n-ekre (pirossal, alul s_n ill. S_n)

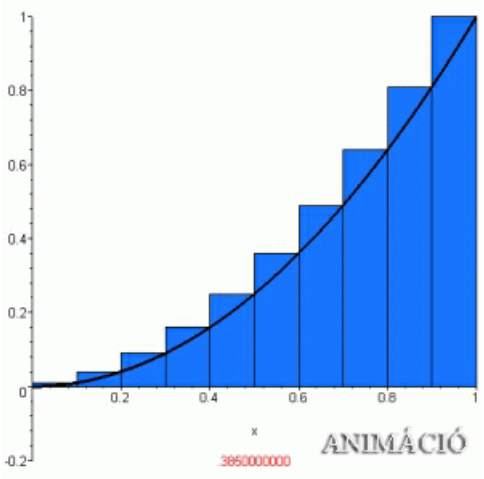
n=5



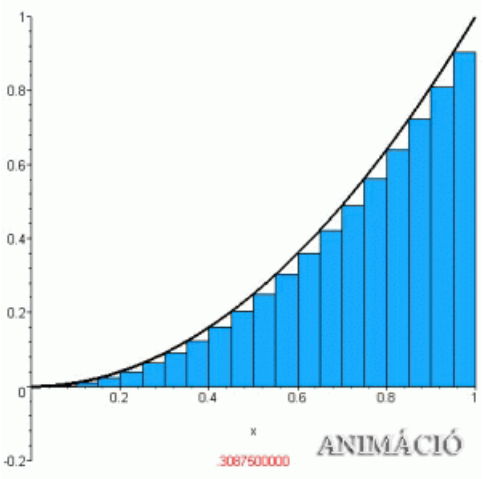


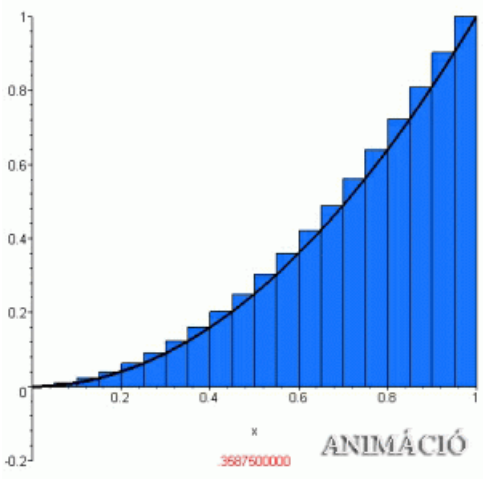
n=10



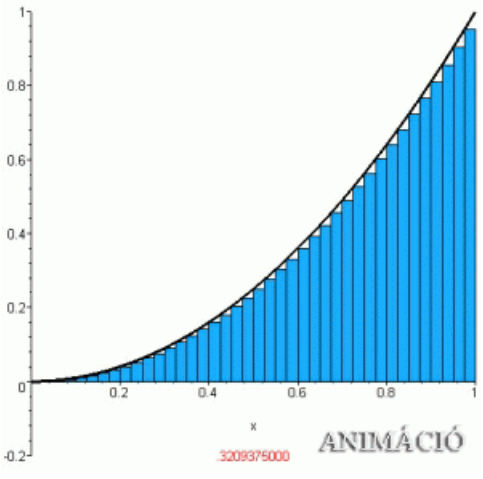


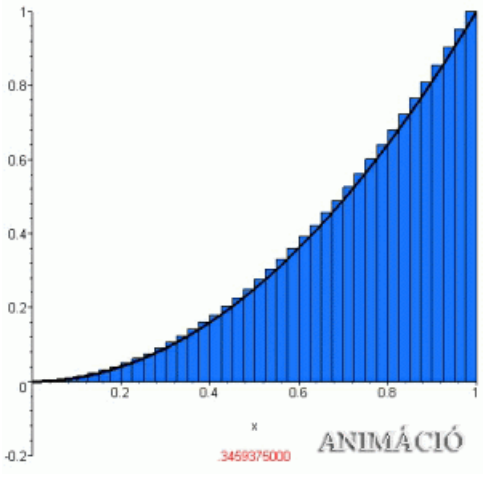
n=20



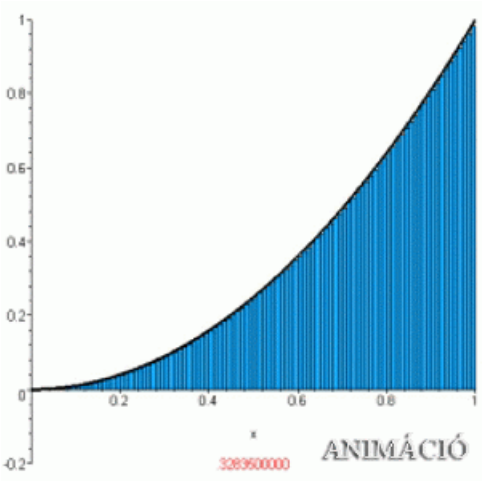


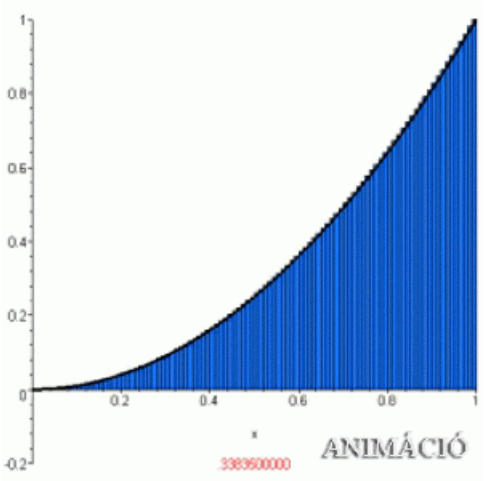
n=40





n=100


1
0.8
0.6
0.4
0.2
0
-0.2
0.2
0.4
0.6
0.8
1
x
ANIMÁCIÓ


1
0.8
0.6
0.4
0.2
0
-0.2
0.2
0.4
0.6
0.8
1
x
ANIMÁCIÓ

Függvények

A következő függvényeknek nézzük meg az alsó illetve felső közelítését:

Alapfüggvények:

- **Négyzetgyökfüggvény**
- **Exponenciális függvény**
- **Logaritmusfüggvény**
- **Szinusz függvény**
- **Koszinusz függvény**

Továbbá néhány, érdekes függvény

Az alább **található eljárások saját munkám eredményei, alapul a The University of Waterloo által bejegyzett leftbox, rightbox és middlebox eljárások szolgáltak.**

Alsó közelítés

```
> with(plots):with(student):

> alsok_szamol:=proc(F, rng, hany)
  local x, i, n, a, b, f,ertek,kezdo,width, parsedargs, ossz;
  parsedargs := `student/checkboxargs`(args);
  x := parsedargs[1];
  a := parsedargs[2];
  b := parsedargs[3];
  n := hany;
  if not type(F, procedure) then f := unapply(F, x)
  else f := F
  end if;
   width := (b - a)/n;
  ossz:=0;
  kezdo:=a;
  for i from 1 by 1 to n
  do
  ertek:=evalf(width*(minimize(f(x),x=kezdo..kezdo+width)));
  ossz:=ossz+ertek;
  kezdo:=kezdo+width;
  end do;
  return ossz;
  end proc:
```

Ez az eljárás az s_n-t számolja ki.

```
> alsok:=proc(F, rng, hany)
   local x, i, n, a, b, f, shadecolor, linecolor, mbox, mboxes,p, t, width, parsedargs, plotopts, smax;
  parsedargs := `student/checkboxargs`(args);
```

```
  x := parsedargs[1];
  a := parsedargs[2];
  b := parsedargs[3];
  n := hany;
  plotopts := op(parsedargs[5]);
  if not type(F, procedure) then f := unapply(F, x)
  else f := F
  end if;
  shadecolor := select(has, [plotopts], 'shading');
  if shadecolor <> [] then shadecolor := `plot/color`(subs(shadecolor, 'shading'))
 else shadecolor := COLOR(HUE,0.56)
 end if;
 p := NULL;
 for t in [plotopts] do
 if lhs(t) <> 'shading' then p := p, t end if
 end do;
 plotopts := p;
 linecolor := select(has, [plotopts], {colour, color});
 if nops(linecolor) = 2 then linecolor := linecolor[2]
 elif nops(linecolor) = 1 then linecolor := linecolor[1]
 else linecolor := colour = black
 end if;
 mbox := subs('_COLOR' = shadecolor, (f, a, b) -> POLYGONS(
 evalf([[a, 0], [a, minimize(f(x),x=a..b)],[b, minimize(f(x),x=a..b)], [b, 0]]),_COLOR));
 width := (b - a)/n;
 mboxes := PLOT(seq(mbox(f, a + i*width, a + (i + 1)*width),i = 0 .. n - 1));
 Felir:=textplot([(0.5,-0.2),alsok_szamol(f,x=a..b,n)],color=red);
 plots[display]({mboxes, Felir, plot(f(x), x = a .. b,thickness = 3, linecolor, 'style' = 'LINE', plotopts)})
 end proc:
```

Ez az eljárás az alsó közelítést szemlélteti, berajzolja az adott finomítás mértékének megfelelően a téglalapokat a függvénygörbéhez.

```
> abrazol_also := proc(f,k,v,p)
  local fv,kezdo,veg,parc,i,Abra:
  fv:=f;
  kezdo:=k;
  veg:=v;
  parc:=p;
  for i from 1 to parc
  do
  Abra[i]:=alsok(fv,x=kezdo..veg,i)
  end do:
  display([seq(Abra[i],i=1..parc)],scaling=unconstrained,insequence=true);
  end proc:
```

Az előző két eljárás egy animációban való ábrázolását készíti el.

Felső közelítés:

```
> with(plots):with(student):

> felsok_szamol:=proc(F, rng, hany)
  local x, i, n, a, b, f,ertek,kezdo,width, parsedargs, ossz;
  parsedargs := `student/checkboxargs`(args);
  x := parsedargs[1];
  a := parsedargs[2];
  b := parsedargs[3];
  n := hany;
  if not type(F, procedure) then f := unapply(F, x)
  else f := F
  end if;
  width := (b - a)/n;
  ossz:=0;
  kezdo:=a;
  for i from 1 by 1 to n
  do
  ertek:=evalf(width*(maximize(f(x),x=kezdo..kezdo+width)));
  ossz:=ossz+ertek;
  kezdo:=kezdo+width;
  end do;
  return ossz;
  end proc:
```

Ez az eljárás az S_n-t számolja ki.

```
> felsok:=proc(F, rng, hany)
  local x, i, n, a, b, f, shadecolor, linecolor, mbox, mboxes,p, t, width, parsedargs, plotopts, smax, Felir;
  parsedargs := `student/checkboxargs`(args);
  x := parsedargs[1];
  a := parsedargs[2];
  b := parsedargs[3];
  n := hany;
  plotopts := op(parsedargs[5]);
  if not type(F, procedure) then f := unapply(F, x)
  else f := F
  end if;
  shadecolor := select(has, [plotopts], 'shading');
  if shadecolor <> [] then shadecolor := `plot/color`(subs(shadecolor, 'shading'))
  else shadecolor := COLOR(HUE, 0.6)
  end if;
  p := NULL;
  for t in [plotopts] do
  if lhs(t) <> 'shading' then p := p, t end if
  end do;
```

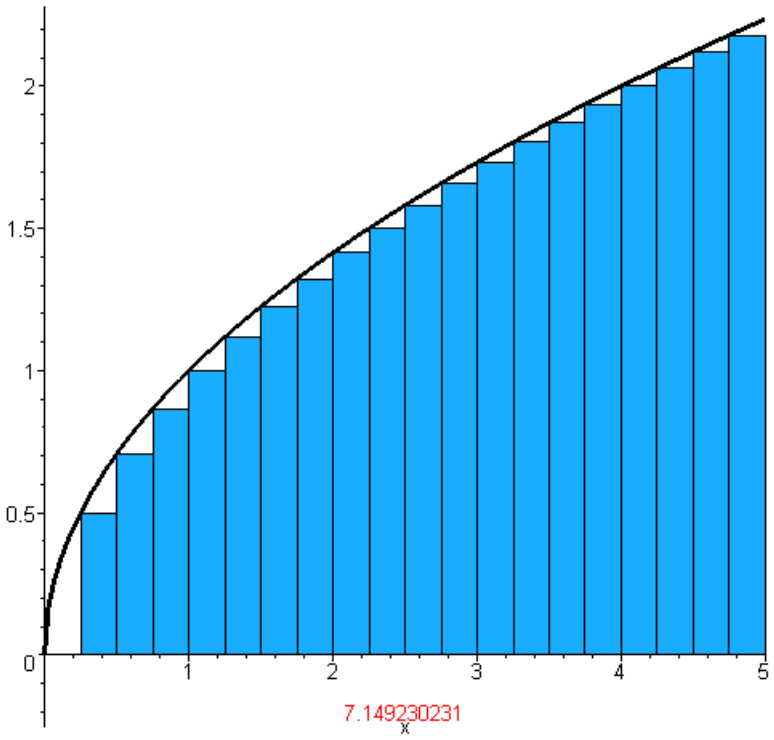
```
  plotopts := p;
  linecolor := select(has, [plotopts], {colour, color});
  if nops(linecolor) = 2 then linecolor := linecolor[2]
  elif nops(linecolor) = 1 then linecolor := linecolor[1]
  else linecolor := colour = black
  end if;
  mbox := subs('_COLOR' = shadecolor, (f, a, b) -> POLYGONS(
  evalf([[a, 0], [a, maximize(f(x),x=a..b)],[b, maximize(f(x),x=a..b)], [b, 0]]),_COLOR));
  width := (b - a)/n;
  mboxes := PLOT(seq(mbox(f, a + i*width, a + (i + 1)*width),i = 0 .. n - 1));
  Felir:=textplot([(0.5,-0.2),felsok_szamol(f,x=a..b,n)],color=red);
  plots[display]({mboxes, Felir, plot(f(x), x = a .. b,thickness = 3, linecolor, 'style' = 'LINE', plotopts)})
  end proc:
```

Ez az eljárás a felső közelítést szemlélteti, berajzolja az adott finomítás mértékének megfelelően a téglalapokat a függvénygörbéhez.

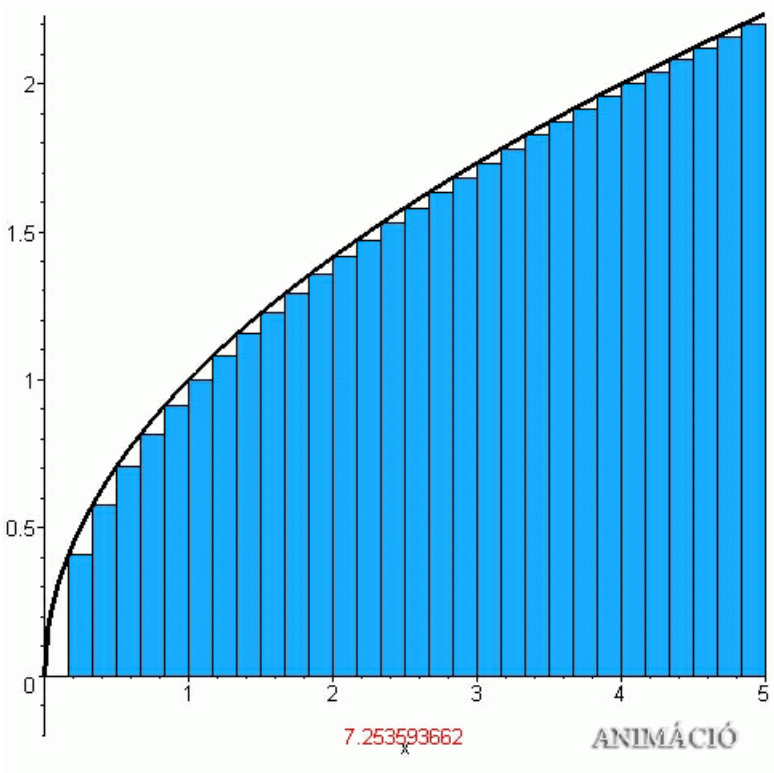
```
> abrazol_felso := proc(f,k,v,p)
  local fv,kezdo,veg,parc,i,Abra:
  fv:=f;
  kezdo:=k;
  veg:=v;
  parc:=p;
  for i from 1 to parc
  do
  Abra[i]:=felsok(fv,x=kezdo..veg,i)
  end do:
  display([seq(Abra[i],i=1..parc)],scaling=unconstrained,insequence=true);
  end proc:
```

Az előző két eljárás egy animációban való ábrázolását készíti el.

> **alsok(sqrt(x),x=0..5,20);**

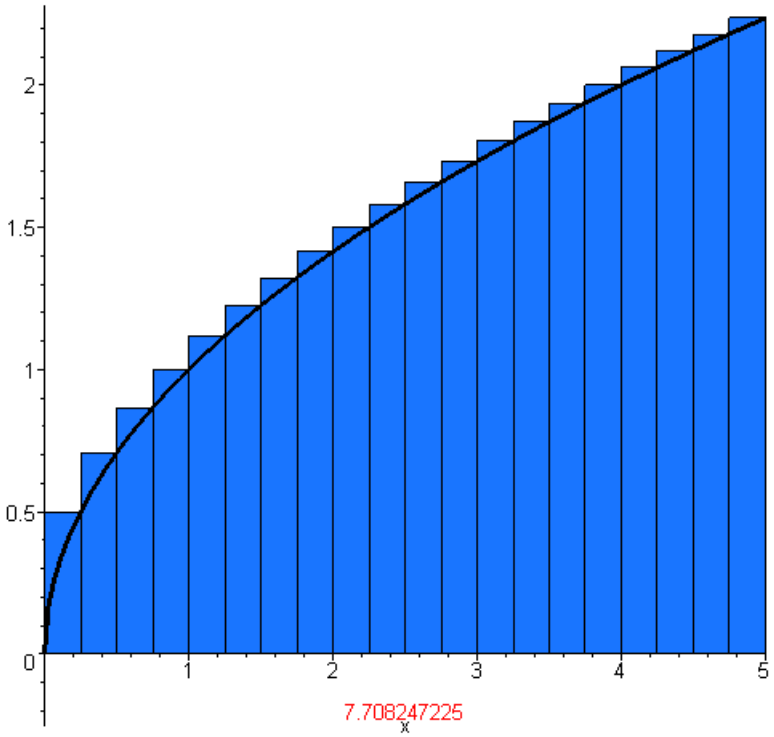


> **abrazol_also(sqrt(x),0,5,30);**

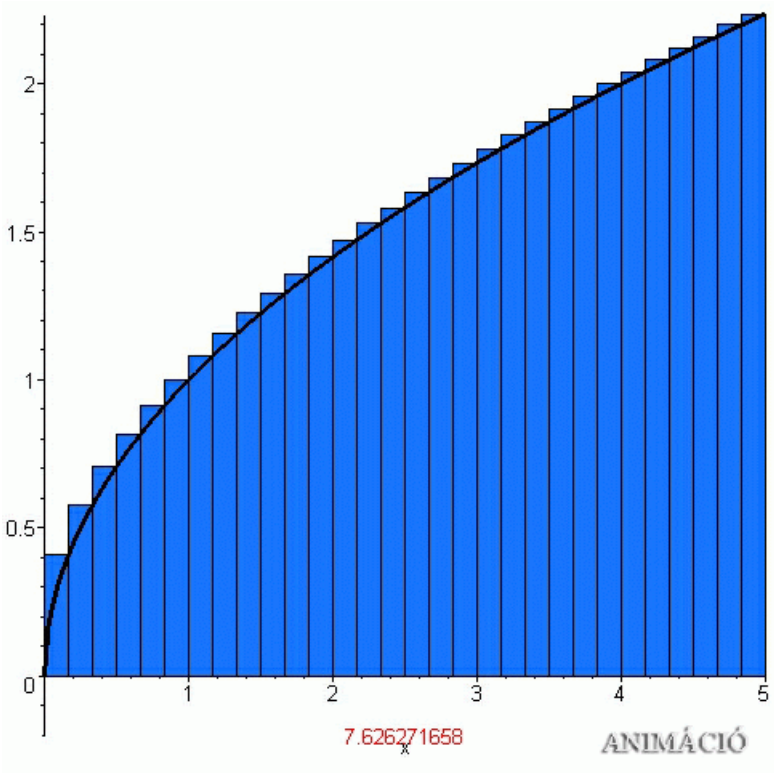


>

felsok(sqrt(x),x=0..5,20);

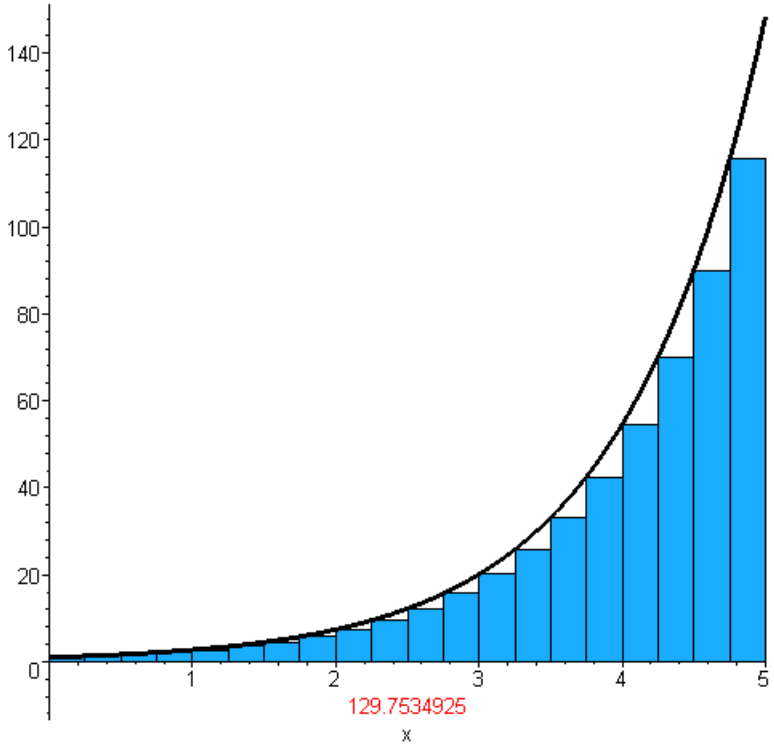


> **abrazol_felso(sqrt(x),0,5,30);**

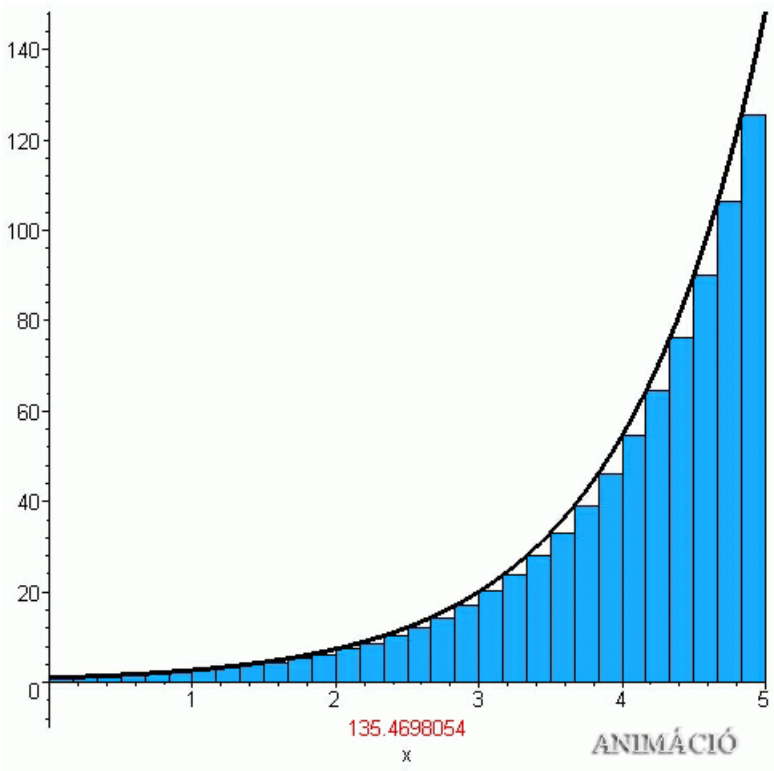


>

> **alsok(exp(x),x=0..5,20);**

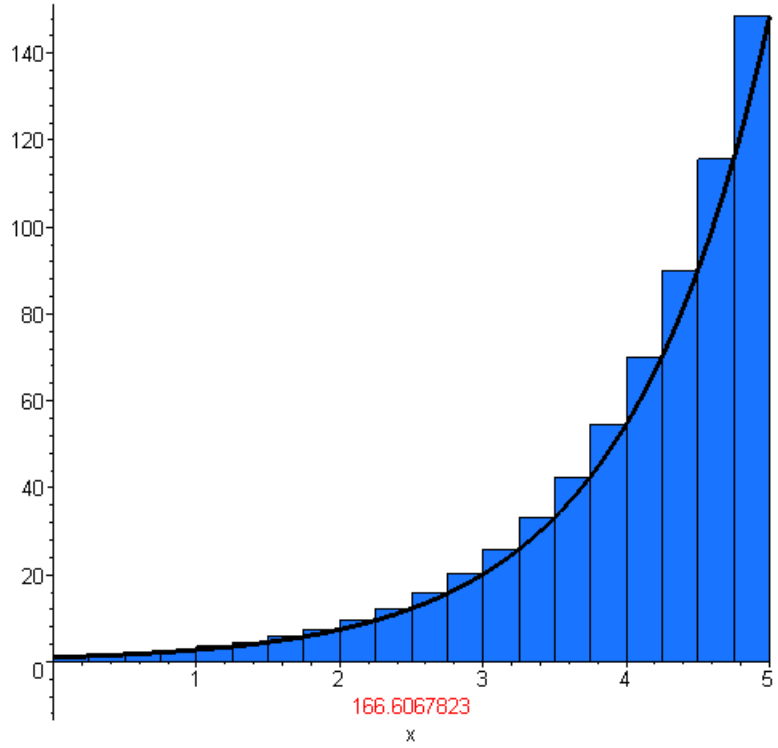


> **abrazol_also(exp(x),0,5,30);**



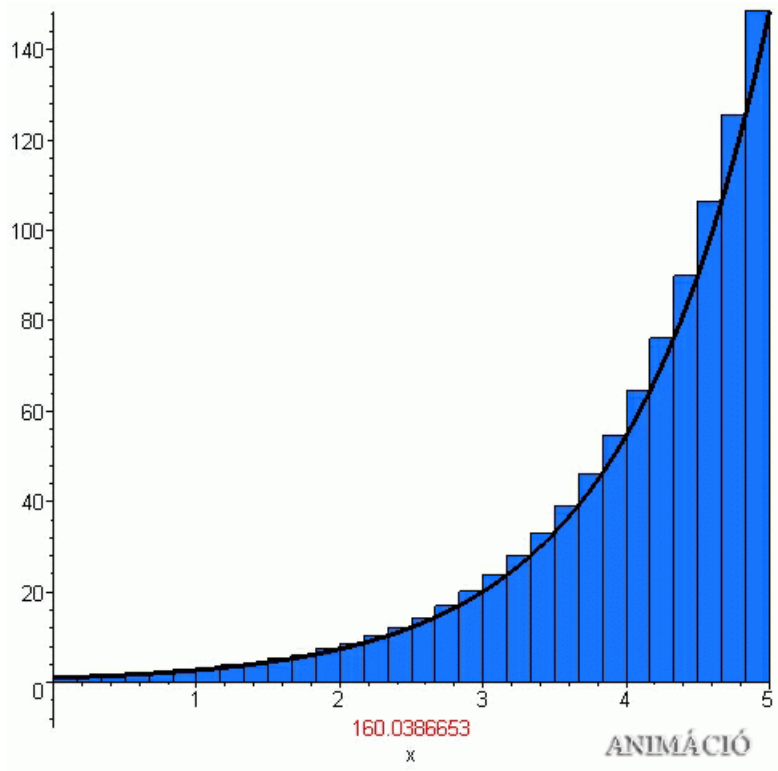
>

> **felsok(exp(x),x=0..5,20);**



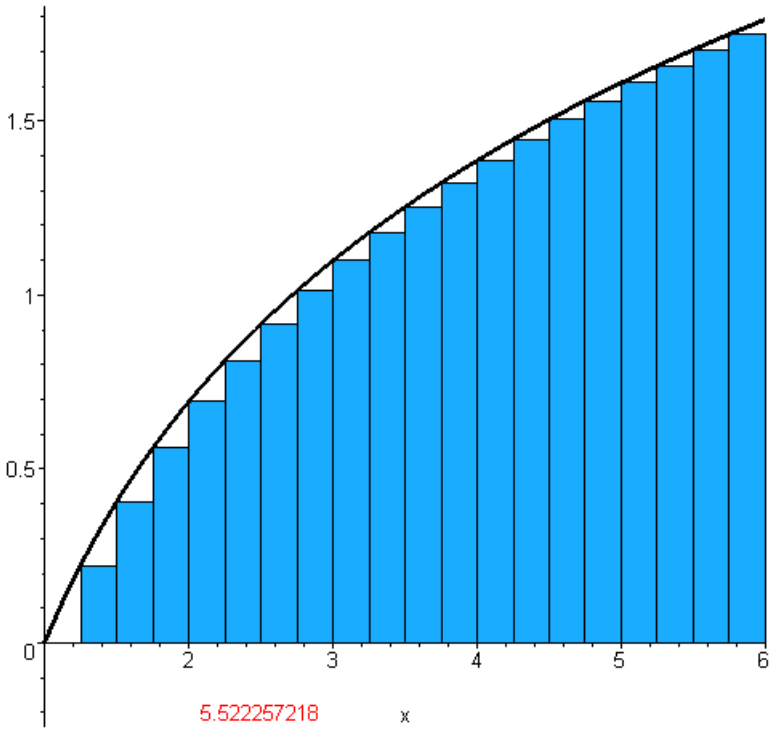
> **abrazol_felso(exp(x),0,5,30);**

>

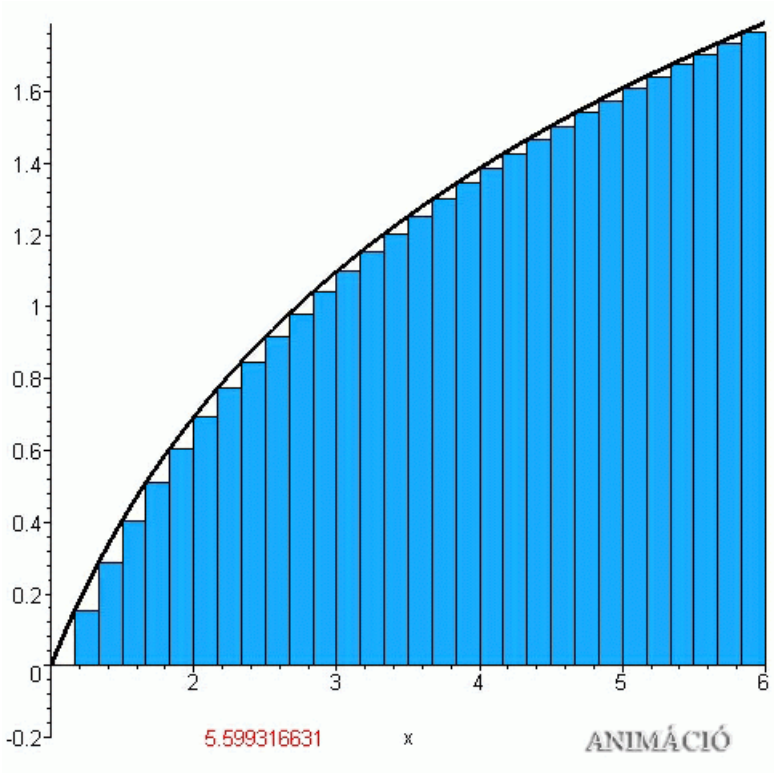


>

> **alsok(log(x),x=1..6,20);**

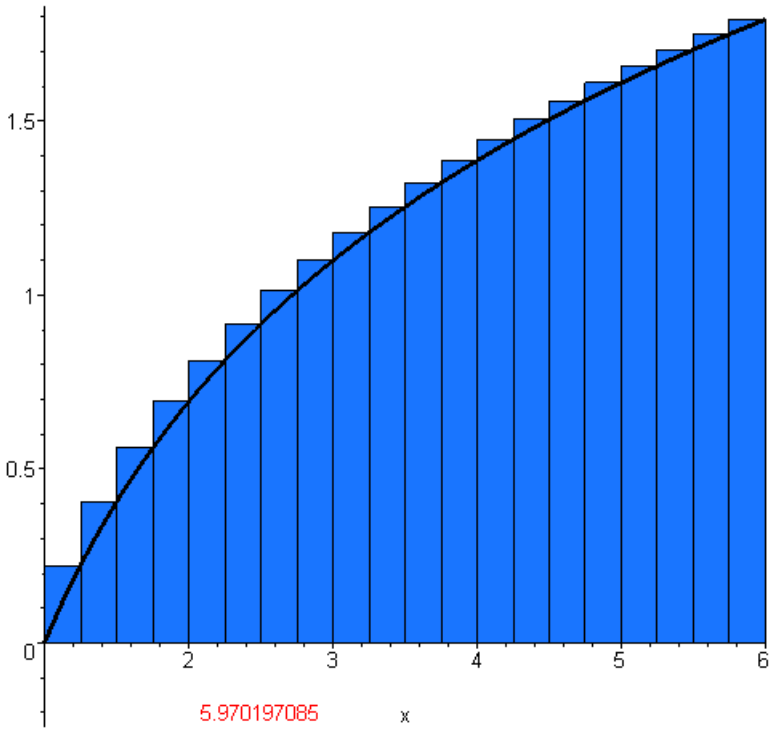


> **abrazol_also(log(x),1,6,30);**

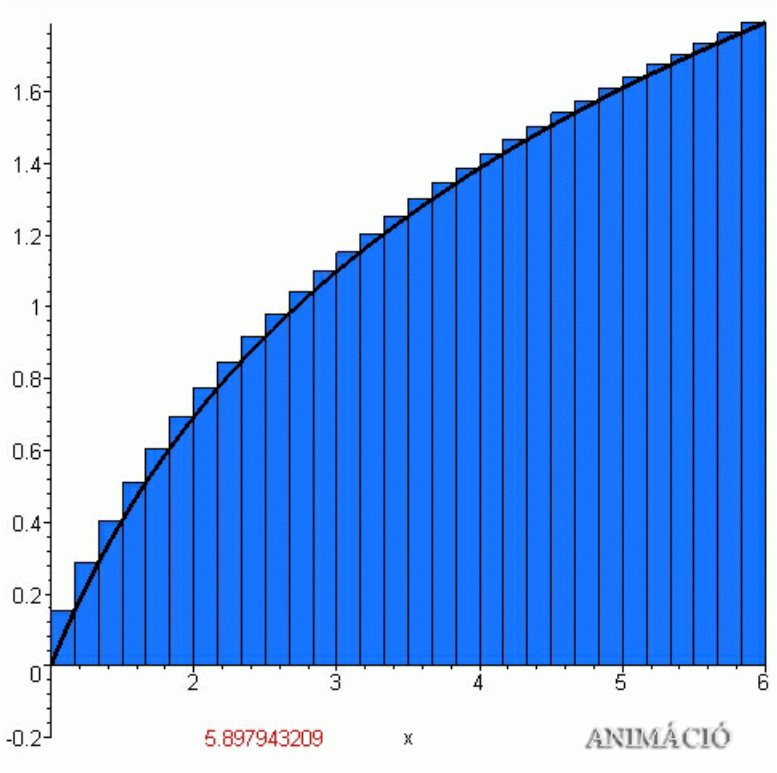


>

> **felsok(log(x),x=1..6,20);**

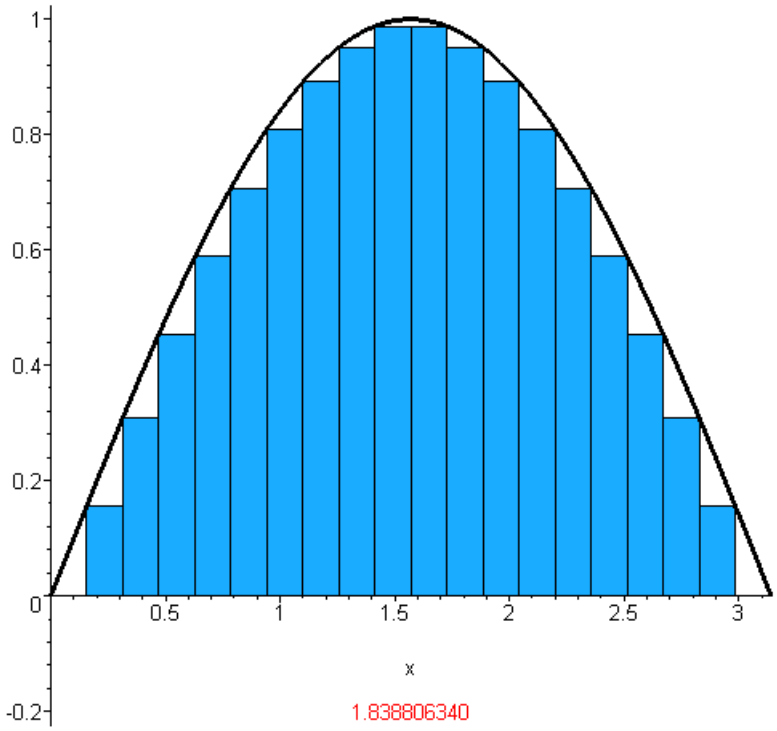


> **abrazol_felso(log(x),1,6,30);**

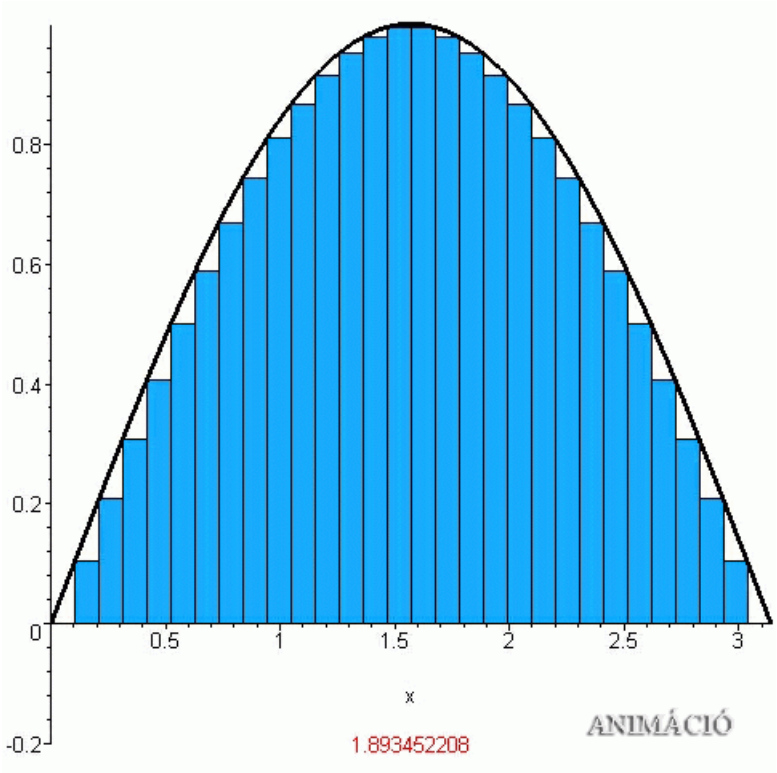


>

> **alsok(sin(x),x=0..Pi,20);**

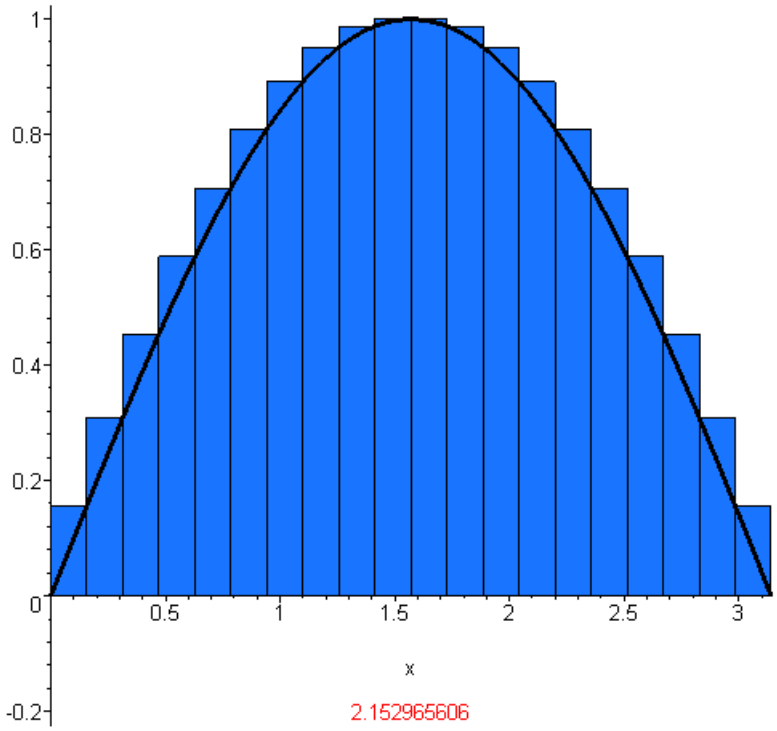


> **abrazol_also(sin(x),0,Pi,30);**

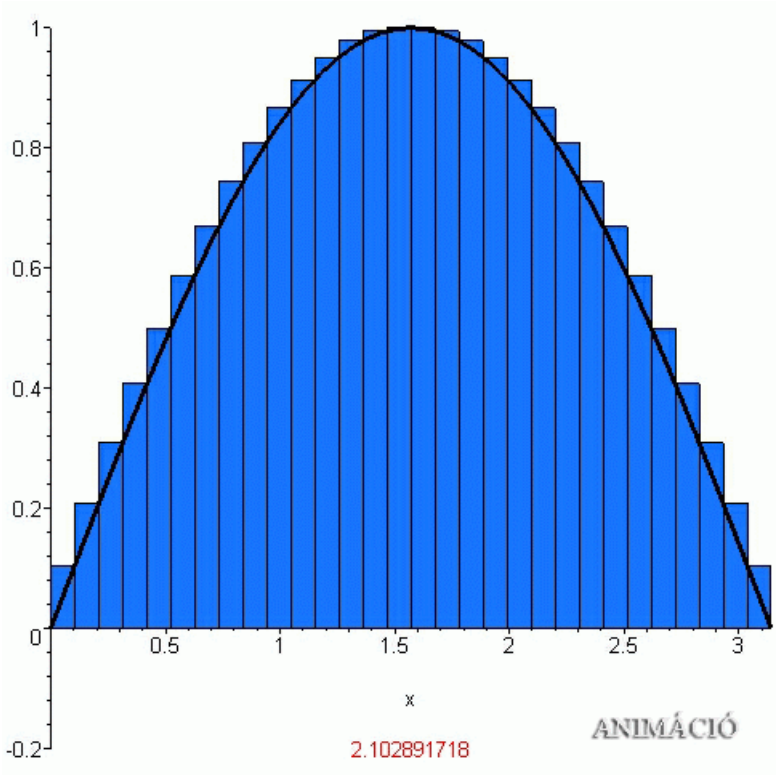


>

> **felsok(sin(x),x=0..Pi,20);**

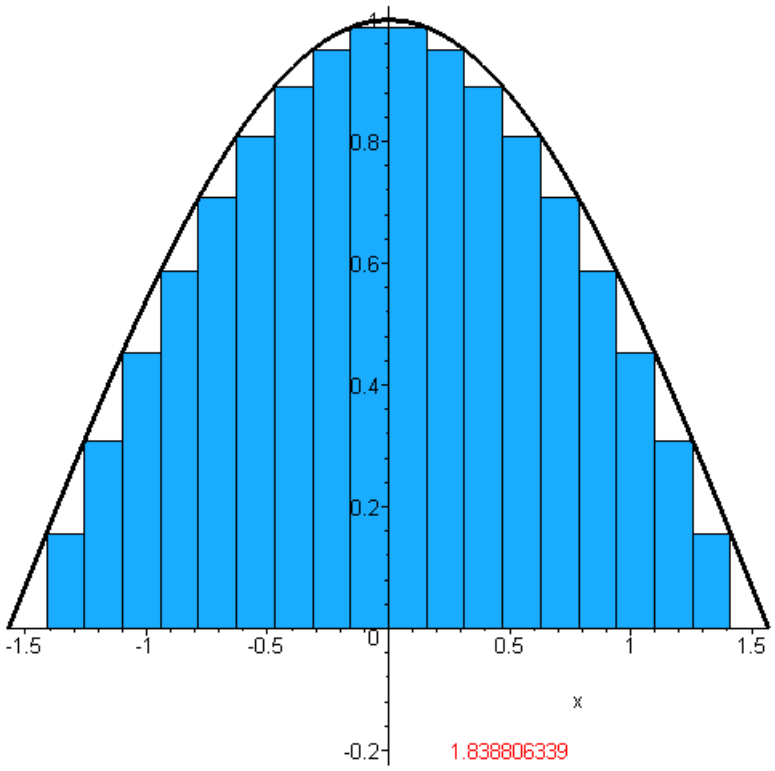


> **abrazol_felso(sin(x),0,Pi,30);**

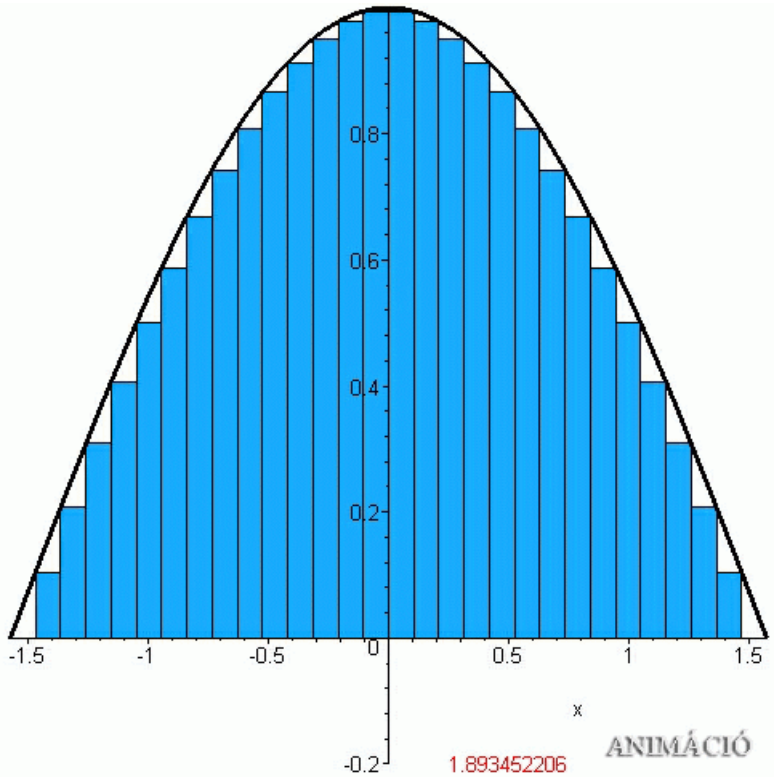


>

> **alsok(cos(x),x=-Pi/2..Pi/2,20);**

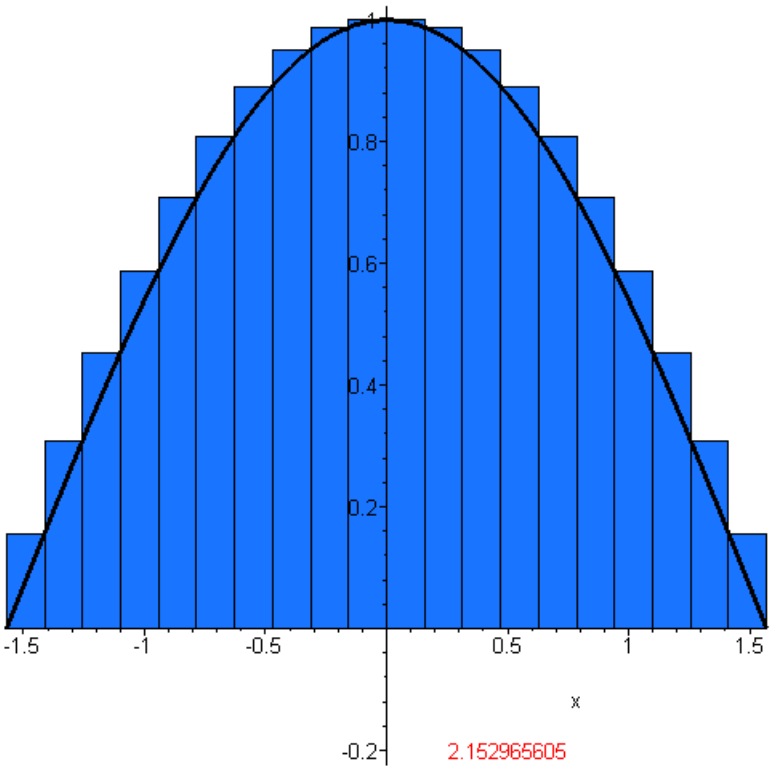


> **abrazol_also(cos(x),-Pi/2,Pi/2,30);**

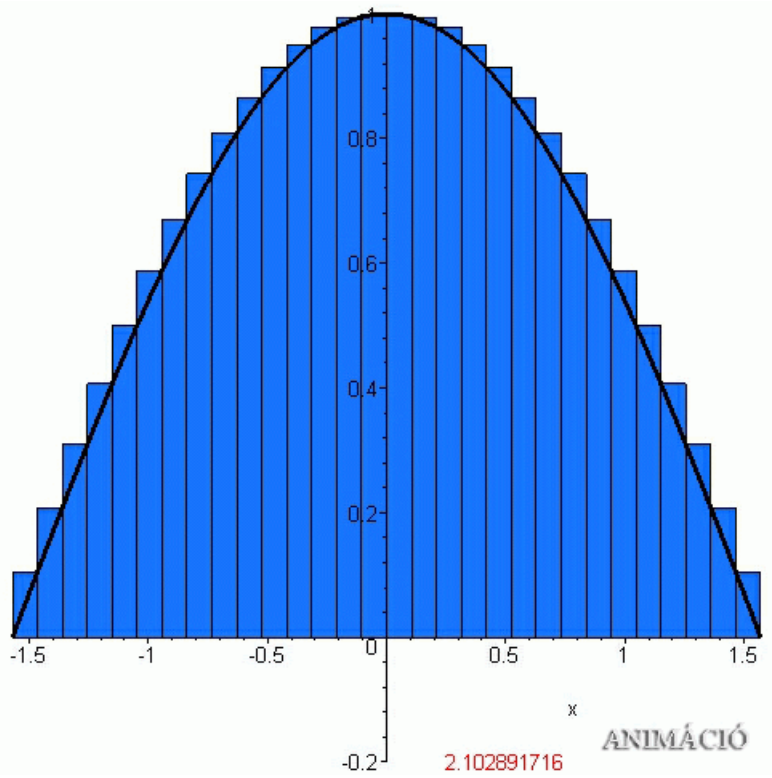


>

> **felsok(cos(x),x=-Pi/2..Pi/2,20);**

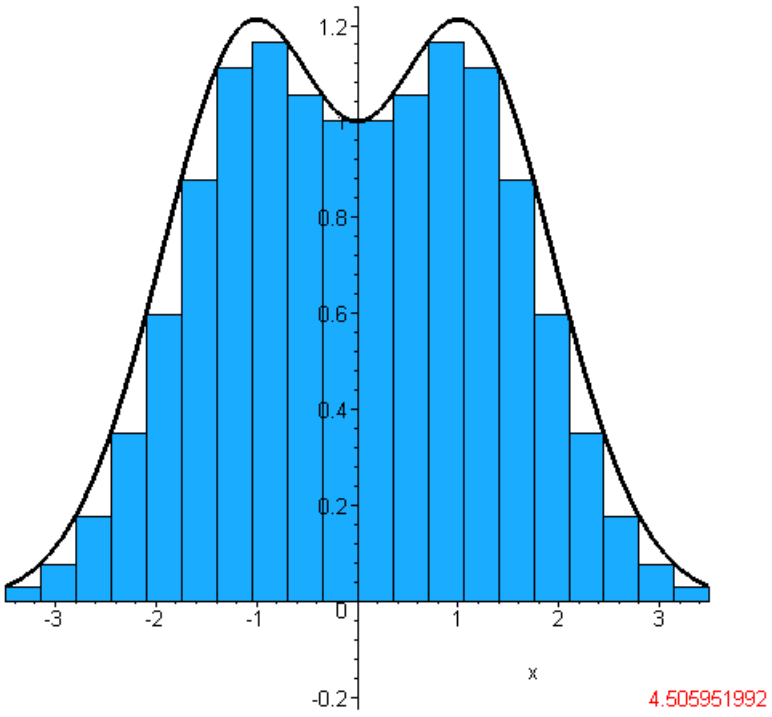


> **abrazol_felso(cos(x),-Pi/2,Pi/2,30);**

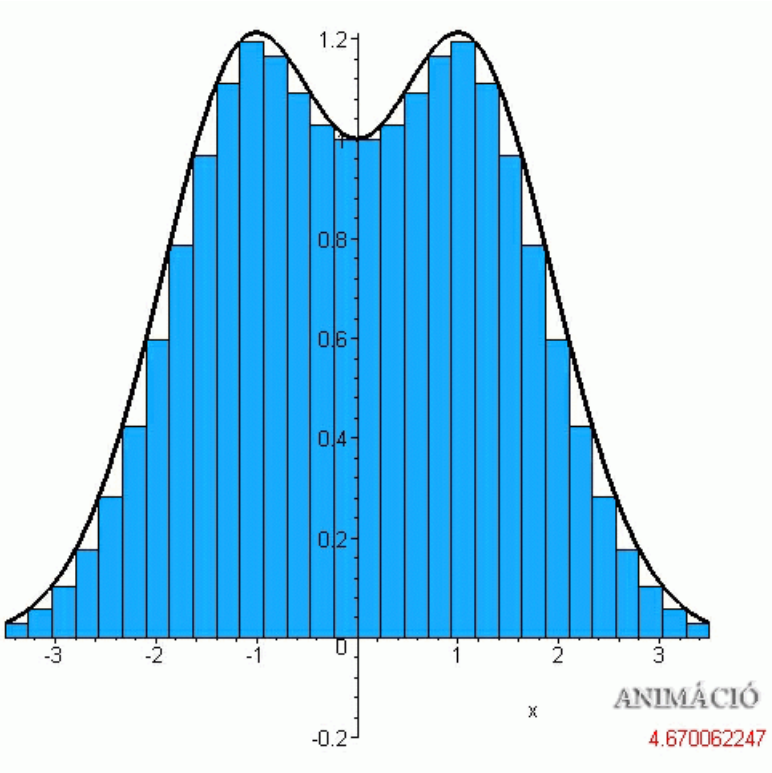


>

> **alsok((1+x^2)*exp(-x^2/2),x=-3.5..3.5,20);**

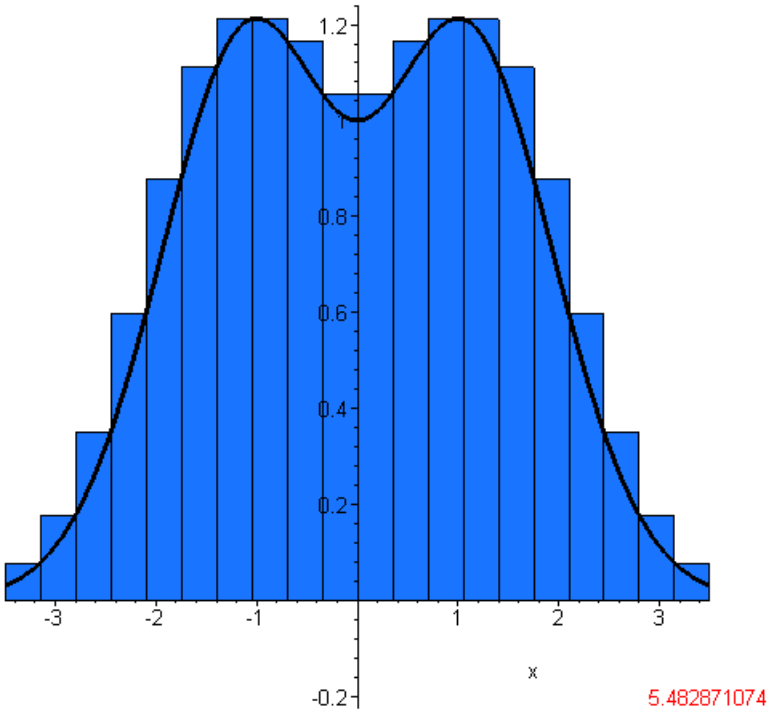


> **abrazol_also((1+x^2)*exp(-x^2/2),-3.5, 3.5,30);**

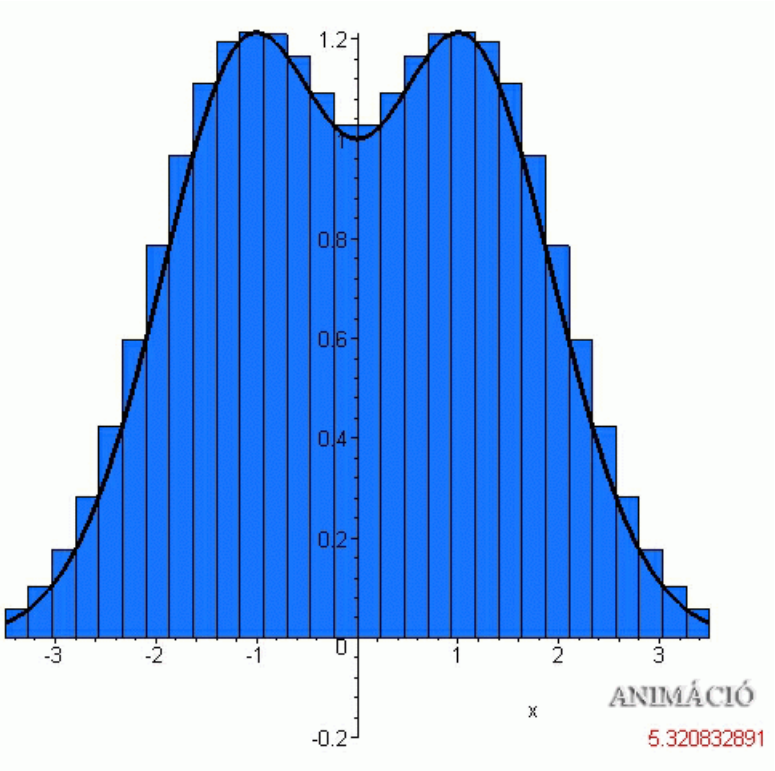


>

felsok((1+x^2)*exp(-x^2/2),x=-3.5..3.5,20);



> **abrazol_felso((1+x^2)*exp(-x^2/2),-3.5, 3.5,30);**



>

A határozott integrál fogalma

Az előzőekben kiszámítottuk az f: R→R, $f(x)=x^2$ függvény görbéje alatti területet a [0;1] intervallumban. Közben megismerkedtünk az analízis egyik alapvető módszerével, a kétoldali közelítés módszerével. A kétoldali közelítésben szerepet játszott a beírt és körülírt téglalapok területének összege. Ezeket s_n-el és S_n-el jelöltük. Nevezzük a továbbiakban s_n-et alsó és az S_n-et felső összegnek.

Általánosítsunk:

Legyen az f: R→R, x → f(x) függvény folytonos az [a;b] intervallumon (a < b) és legyen f(x) ≥ 0. Alkalmazzuk a parabolikus háromszög területének kiszámításánál követett módszert, és számítsuk ki a görbe alatti terület az [a;b] intervallumban.

Pontosabban:

Határozzuk meg annak a síkidomnak a területét, amelyet felülről az y=f(x) egyenletű folytonos görbe vonal, alulról az x tengely és oldalról az a és b abszcisszájú görbepontok ordinátái (a görbepontokból az x tengelyre húzott merőleges szakaszok) határolnak.

Osszuk fel az [a;b] intervallumot n egyenlő részre. Jelöljük a részintervallumok kezdő, illetve végpontjainak abszcisszáit $x_0,x_1,x_2,...x_n$-nek, ahol x_0=a és x_n=b. Jelöljük ki az $[x_0,x_1]$ intervallumon a legkisebb függvényértéket m_1-gyel, a legnagyobb függvényértéket M_1-gyel, az $[x_1,x_2]$ intervallumon a legkisebb függvényértéket m_2-vel, a legnagyobb függvényértéket M_2-vel és így tovább. Az $[x_{n-1},x_n]$ intervallumon a legkisebb függvényérték legyen m_n, a legnagyobb függvényérték legyen M_n.

Mivel az [a;b] intervallumot n egyenlő részre osztottuk, ezért egy-egy intervallum hossza $\frac{b-a}{n}$.

Írjuk fel az alsó és a felső összegeket:

$$s_n = \frac{b-a}{n}m_1 + \frac{b-a}{n}m_2 + \ldots + \frac{b-a}{n}m_n$$

illetve

$$s_n = \frac{b-a}{n}(m_1 + m_2 + \ldots + m_n)$$

Hasonlóképpen

$$S_n = \frac{b-a}{n}(M_1 + M_2 + \ldots + M_n)$$

A szemléletre is hivatkozva kijelenthetjük, hogy minden n-re bármelyik alsó összeg kisebb (nem nagyobb) bármelyik felső összegnél, azaz $s_n \le S_n$.

Bebizonyítható, hogy az f függvény folytonossága elégséges, de nem szükséges feltétel ahhoz, hogy az alsó és a felső összegeknek az [a;b] intervallumban létezzen közös határértéke: I.

$$\lim_{n\to\infty} s_n = \lim_{n\to\infty} S_n = I$$

I az egyetlen olyan szám, amely a $\{s_n;S_n\}$ intervallumsorozat közös pontja.

Ezt az I számot tekintjük a folytonos nemnegatív f függvény görbéje alatti terület mérőszámának az [a;b] intervallumon.
A fenti megoldásnál feltételeztük, hogy az [a;b] intervallumot egyenlő részekre osztottuk, és a felosztást finomítottuk, azaz az n értékét "minden határon túl" növeltük.

Ha az f függvény az [a;b] intervallumban folytonos, akkor az intervallum tetszőleges felosztását tekintve az alsó és a felső összegeknek pontosan egy közös határértéke van. Feltéve, hogy a felosztást úgy finomítjuk, hogy a leghosszabb részintervallum hossza 0-hoz tart. Az így kapott határérték azonos az egyenlő felosztással adódó alsó és felső összegek közös határértékével.

A kapott $\lim_{n\to\infty} s_n = \lim_{n\to\infty} S_n = I$ **számot az f folytonos függvény Riemann-féle határozott integráljának nevezzük.**

A határozott integrált a következőképpen jelöljük: $\int_a^b f$ **vagy** $\int_a^b f(x)dx$ **.**

Megjegyzés: A határozott integrál fogalma általában nem csak folytonos függvények esetén definiálhatóak, de jelen dolgozatban az egyszerűség kedvéért csak ezekre szorítkozom.

Integrálszámítás a felsőbb matematikában

A differenciálszámításhoz hasonlóan a középiskolai tárgyalásmód mellett, aki szeretné megtekinteni, tanulmányozni az integrálszámítás magasabb szinten való tárgyalását az Dr. Lajkó Károly, egyetemi docens Analízis 2 jegyzetéből megteheti.

A jegyzet az alábbi linken érhető el: http://www.math.klte.hu/~lajko/

Irodalomjegyzék

[1] Bárczy Barnabás: Differenciálszámítás, Műszaki Könyvkiadó, 1997

[2] Bárczy Barnabás: Integrálszámítás, Műszaki Könyvkiadó, 1997

[3] Czapáry Endre - Gyapjas Ferenc: Matematika a középiskolák 11.-12. évfolyama számára Nemzeti Tankönyvkiadó, 2002

[4] Hajnal Imre - Dr. Pintér Lajos: Matematika III. (fakultatív B változat), Nemzeti Tankönyvkiadó, 1999

[5] Lajkó Károly: Analízis II., egyetemi jegyzet, Kossuth Egyetemi Kiadó, 2003

[6] Molnárka Győző - Gergó Lajos - Wettl Ferenc - Horváth András - Kallós Gábor: A Maple V. és alkalmazásai Springer Hungarica Kiadó Kft., 1996

[7] Obádovics J. Gyula: Matematika, Scolar Kiadó, 1994, Tizenötödik kiadás

[8] Obádovics J. Gyula - Szarka Zoltán: Felsőbb Matematika, Scolar Kiadó, 2002, Második, javított kiadás

Köszönetnyilvánítás

A következő személyeknek szeretnék köszönetet mondani szakdolgozatom elkészülésében nyújtott közvetlen vagy közvetett segítségükért:

- **Elsősorban szüleimnek, testvéreimnek (kiemelten Anna nővéremnek) és családom minden tagjának, akik lehetőséget adtak arra, hogy a Debreceni Egyetem hallgatója lehessek. Köszönet a bizalomért, az állandó szellemi és anyagi támogatásért**
- **Dr. Gilányi Attilának témám vezetéséért, az ötletekért, segítségéért, javításaiért**
- **Dr. Lajkó Károlynak, aki bevezetett igazi tanárként az analízis rejtelmeibe**
- **A Maple program fejlesztőinek, akik megalkották és folyamatosan fejlesztik ezt a nagyon hasznos és csodás alkalmazást**

Tartalomjegyzék

BEVEZETÉS **2**

BEVEZETÉS A DIFFERENCIÁLSZÁMÍTÁSBA **5**

Az érintő szemléletes fogalma **5**

Az érintő definiálása **7**

Függvények és érintőik **9**

Egy Ön által választott függvény **20**

A differenciálszámítás a felsőbb matematikában **26**

BEVEZETÉS AZ INTEGRÁLSZÁMÍTÁSBA **27**

Kétoldali közelítés **27**

Függvények **31**

A határozott integrál fogalma **47**

Integrálszámítás a felsőbb matematikában **48**

IRODALOMJEGYZÉK **49**

KÖSZÖNETNYILVÁNÍTÁS **50**