Alkalmazott matematika

Baran Ágnes

Nemlineáris egyenletek

Nemlineáris egyenletek

Az $f(x) = 0$ egyenlet gyökeit keressük, ahol $f : \mathbb{R} \to \mathbb{R}$ nemlineáris függvény.

Példa:

$$\cos(x) - x = 0$$

vagy

$$x^5 - 3x^4 + x^3 - 5x^2 + 3 = 0$$

vagy

$$e^x - 4x^2 = 0$$

vagy

$$\ln(x) - x + 2 = 0$$

A gyök numerikus közelítése

Az $f(x) = 0$ egyenlet gyökét egy $\{x_k\}$, $k = 0, 1, 2, ...$ sorozattal (iteráció) fogjuk közelíteni.

A közelítés adott, ha adott

- az x_0 kiindulópont,
- az algoritmus x_{k+1} meghatározására, ha x_k már ismert,
- a leállási feltétel.

1. Felezési módszer

Tfh $f : [a, b] \to \mathbb{R}$ folytonos, és $f(a) \cdot f(b) < 0$
Ekkor az

$$f(x) = 0$$

egyenletnek van gyöke (a, b)-ben.

Az algoritmus

Adott a maximális iterációszám (maxit) és az ε pontosság.

1. legyen $k = 1$, $x_0 = a$ és $x_1 = b$
2. legyen $x_2 = \frac{x_0 + x_1}{2}$
3. a) ha $f(x_2) = 0$, akkor x_2 gyök $\to$ kilépés (eredmény: x_2)
   b) ha $f(x_0) \cdot f(x_2) < 0$, akkor $x_1 = x_2$
   c) ha $f(x_1) \cdot f(x_2) < 0$, akkor $x_0 = x_2$

   ha $|x_1 - x_0| < \varepsilon \to$ kilépés (eredmény: x_2)

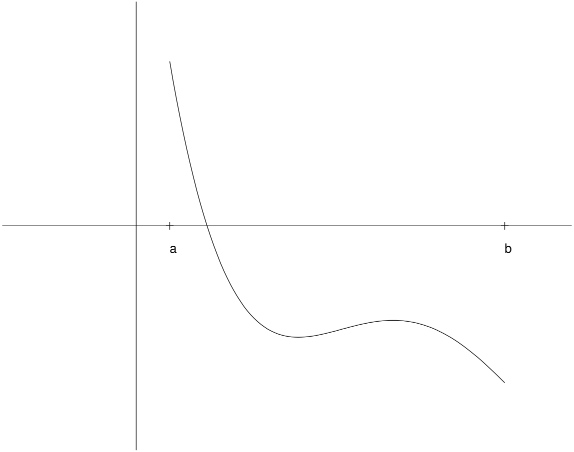
   $k := k + 1$

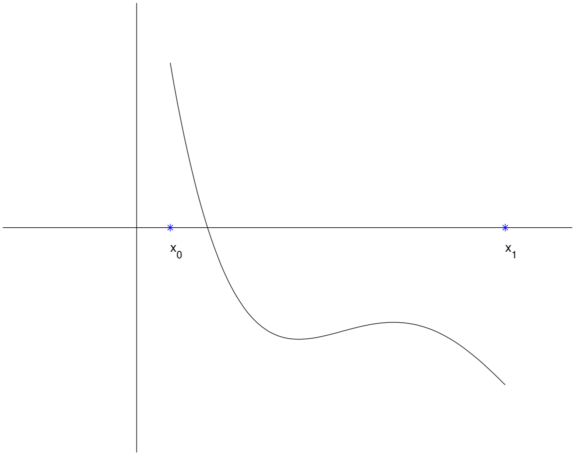
   ha $k = maxit \to$ kilépés (maxit lépésben nem találtunk gyököt)

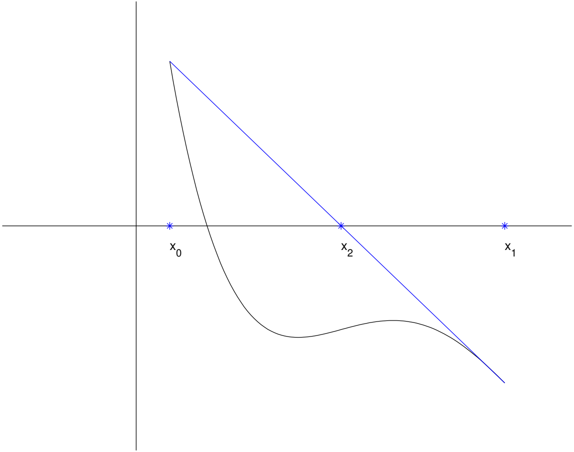
   $\to 2.$

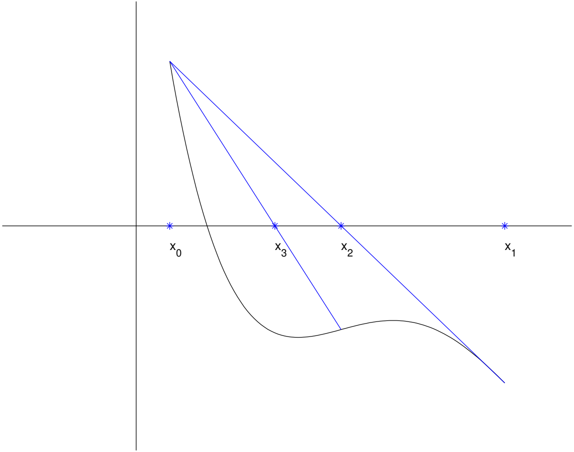
2. Húrmódszer

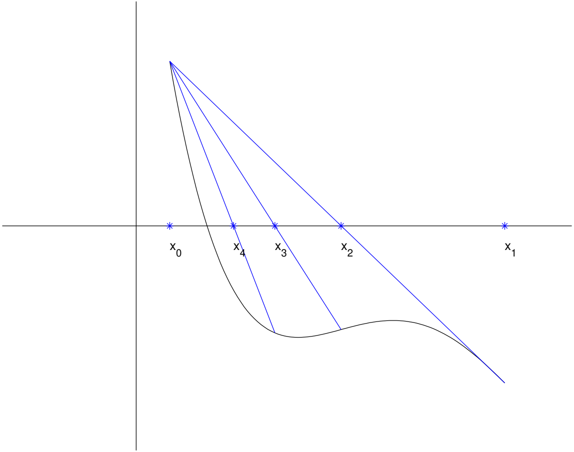
Az $f(x) = 0$ nemlineáris egyenlet gyökét keressük, ahol $f : [a, b] \to \mathbb{R}$, továbbá $f(a) \cdot f(b) < 0$ és f folytonos.



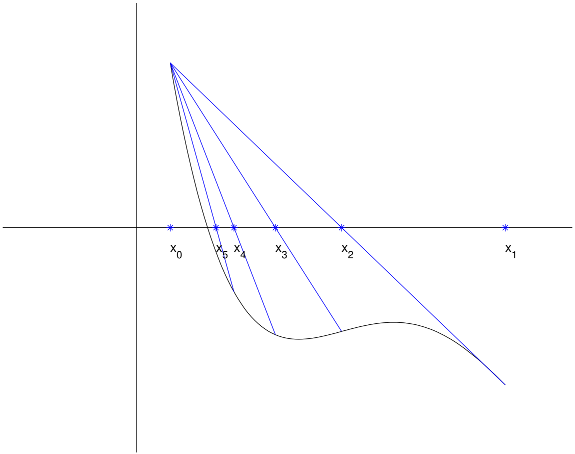

x_0
x_1


x0
x2
x1


x0
x3
x2
x1



x_0
x_4
x_3
x_2
x_1


x_0
x_5
x_4
x_3
x_2
x_1

Húrmódszer

$x_0 = a$, $x_1 = b$.
Az x_2 pont meghatározása:

Az $(x_0, f(x_0))$ és $(x_1, f(x_1))$ pontokra illeszkedő egyenes egyenlete (Lagrange-interpoláció):

$$\begin{array}{c|cc} x_0 & f(x_0) & \\ & & \frac{f(x_1)-f(x_0)}{x_1-x_0} \\ x_1 & f(x_1) & \end{array}$$

$y(x) = f(x_1) + \frac{f(x_1)-f(x_0)}{x_1-x_0} \cdot (x - x_1)$

Ott metszi az x-tengelyt, ahol $y(x) = 0$:

$$x_2 = x_1 - f(x_1) \cdot \frac{x_1 - x_0}{f(x_1) - f(x_0)}$$

Húrmódszer

x_2 kiszámítása után ismételjük meg az előző lépéseket az $[x_0, x_2]$, illetve $[x_2, x_1]$ intervallumok közül azzal, ahol előjelet vált a függvény.

A húrmódszer esetén

- x_2 kiszámítása jól definiált
- az eljárás minden folytonos f esetén konvergál f egy gyökéhez
- csak páratlan multiplicitású gyök közelítésére
- két pontra támaszkodó iteráció

Az algoritmus:

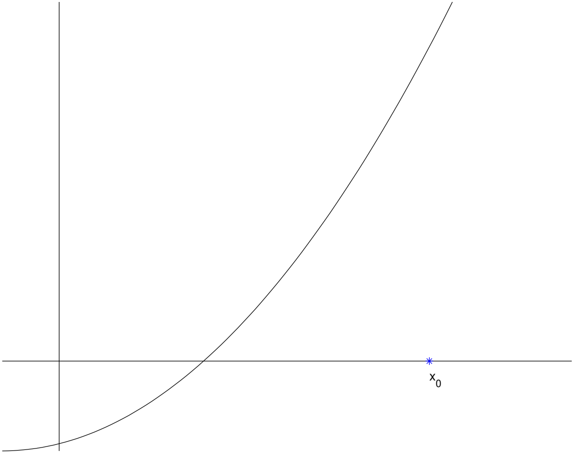
Adott a maximális iterációszám (maxit) és az ε pontosság.

1. $x_0 := a$, $x_1 := b$, $f0 := |f(x_0)|$
2. $$x_2 := x_1 - f(x_1) \cdot \frac{x_1 - x_0}{f(x_1) - f(x_0)}$$
3. a) Ha $f(x_2) = 0$, akkor kilépés (x_2 gyök).
   b) ha $f(x_2) \cdot f(x_1) < 0$, akkor $x_0 = x_1$, $x_1 = x_2$
   c) ha $f(x_2) \cdot f(x_0) < 0$, akkor $x_1 = x_2$

   ha $|f(x_2)| < \varepsilon * (1 + f0)$, akkor kilépés (eredmény: x_2)
   $k := k + 1$
   ha $k = maxit$, akkor kilépés (maxit lépésben nem találtunk gyököt)
   $\rightarrow 2.$

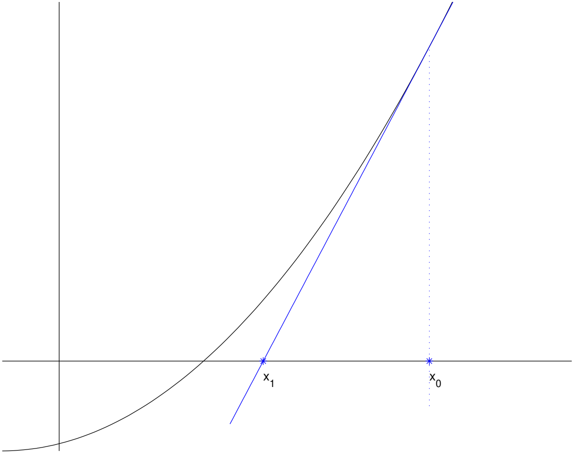
3. Newton-módszer

Az $f(x) = 0$ nemlineáris egyenlet gyökét keressük.



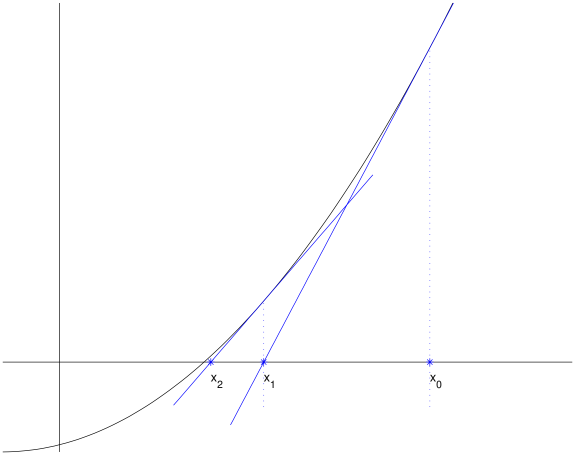
3. Newton-módszer

Az $f(x) = 0$ nemlineáris egyenlet gyökét keressük.



3. Newton-módszer

Az $f(x) = 0$ nemlineáris egyenlet gyökét keressük.



Az algoritmus:

x_0 a gyök egy kezdeti közelítése,
x_{k+1} meghatározása:
Az f függvény x_k-beli érintője (Hermite-interpoláció):

$$\begin{array}{c|cc} x_k & f(x_k) & \\ & & f'(x_k) \\ x_k & f(x_k) & \end{array}$$

$$y(x) = f(x_k) + f'(x_k) \cdot (x - x_k)$$

Ott metszi az x-tengelyt, ahol $y(x) = 0$:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

A Newton-iteráció:

x_0 kezdőpont,

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, \ldots$$

- nem feltétlenül definiált
- egy pontra támaszkodó iteráció

Tétel. Legyen x^* az f egy gyöke. Ha

- f kétszer folytonosan diff.ható,
- $|f'(x)| \geq m_1 > 0$,
- $|f''(x)| \leq M_2$,
- $|x_0 - x^*| < \frac{2m_1}{M_2}$,

akkor a Newton-iteráció jól definiált, $x_k \to x^*$, ha $k \to \infty$, továbbá

$$|x_{k+1} - x^*| \leq C|x_k - x^*|^2$$

Mit jelent a gyakorlatban a

$$|x_{k+1} - x^*| \leq C|x_k - x^*|^2$$

becslés?

Ha valamely k-ra $|x_k - x^*| \approx 0.1$, akkor a sorozat következő néhány tagjának a távolsága a gyöktől kb

$$0.01$$
$$0.0001$$
$$0.00000001$$

A Newton-módszer konvergenciája **kvadratikus**, vagy másodrendű.

1. példa

Közelítsük az $x^3 - 3x - 2 = 0$ egyenlet gyökét Newton–módszerrel az $x_0 = 1.5$ pontból indulva!

$f(x) = x^3 - 3x - 2$ és $f'(x) = 3x^2 - 3$.

$$x_{k+1} = x_k - \frac{x_k^3 - 3x_k - 2}{3x_k^2 - 3}, \quad k = 0, 1, 2, \ldots$$

$$\begin{aligned}
x_0 &= 1.5 \\
x_1 &= 2.33333333333333 \\
x_2 &= 2.\underline{0}5555555555556 \\
x_3 &= 2.\underline{00}194931773879 \\
x_4 &= 2.\underline{00000}252829797 \\
x_5 &= 2.\underline{00000000000}426
\end{aligned}$$

2. példa

Közelítsük $\sqrt{a}$, $(a > 0)$ értékét Newton–módszerrel!

$f(x) = x^2 - a$ és $f'(x) = 2x$. Ekkor

$$x_{k+1} = x_k - \frac{x_k^2 - a}{2x_k} = \frac{1}{2}\left(x_k + \frac{a}{x_k}\right)$$

$a = 5$, $x_0 = 2$ esetén:

$$\begin{aligned}
x_1 &= 2.\underline{2}5\\
x_2 &= 2.\underline{236}11111111111\\
x_3 &= 2.\underline{2360679777}91580\\
x_4 &= 2.\underline{23606797749979}
\end{aligned}$$

3. példa

Közelítsük az $x^3 - 3x + 2 = 0$ egyenlet gyökét Newton–módszerrel az $x_0 = 1.5$ pontból indulva!

$f(x) = x^3 - 3x + 2$ és $f'(x) = 3x^2 - 3$.

$$x_{k+1} = x_k - \frac{x_k^3 - 3x_k + 2}{3x_k^2 - 3}, \quad k = 0, 1, 2, \ldots$$

$$\begin{aligned}
x_0 &= 1.5 \\
x_1 &= 1.26666666666667 \\
x_2 &= 1.13856209150327 \\
x_3 &= 1.\underline{0}7077733565581 \\
x_4 &= 1.\underline{0}3579185227111 \\
&\ldots \\
x_9 &= 1.\underline{00}113136084711
\end{aligned}$$

Hasonlítsuk össze az eredmény az 1. példa eredményével! Bár az egyenlet gyökéhez konvergál a sorozat, de a konvergencia nem kvadratikus. Miért?

A probléma: az 1 kétszeres gyöke f-nek (a konvergenciatétel 2. feltétele nem teljesül).

Ha az

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

iteráció helyett az

$$x_{k+1} = x_k - 2\frac{f(x_k)}{f'(x_k)}$$

iterációt alkalmazzuk:

$$\begin{aligned}
x_0 &= 1.5 \\
x_1 &= 1.\underline{0}3333333333333 \\
x_2 &= 1.\underline{000}18214936248 \\
x_3 &= 1.\underline{00000000}552926
\end{aligned}$$

A Newton-iteráció nem feltétlenül konvergál, ezért fontos, hogy programozásakor az $\{x_k\}$ sorozatot legfeljebb egy megadott `maxit` iterációszámig határozzuk meg.

1. feladat

Írjunk egy Matlab-függvényt, mely megadott x_0 kezdőpont, `maxit` maximális iterációszám, ε pontosság esetén egy adott f függvény gyökének közelítésével és az elvégzett iterációk számával tér vissza, illetve ha az algoritmus nem konvergál, vagy egy Newton-lépés nem definiált, akkor a megfelelő hibaüzenettel.

2. feladat

Alkalmazzuk a Newton-módszert az $f(x) = x^3 - 5x$ függvény gyökének közelítésére az $x_0 = 1$ pontból indulva!

3. feladat (Szorgalmi)

Írjunk 1-1 Matlab-függvényt, mely adott f esetén az f függvény gyökének felező-módszerrel, illetve húrmódszerrel való közelítésével és az elvégzett iterációk számával tér vissza.

4. feladat (Szorgalmi)

Közelítsük az 1. példában adott függvény gyökeit az előző két függvénnyel. Hasonlítsuk össze a konvergencia sebességét a Newton-módszer konvergenciasebességével.

4. Szelőmódszer

A Newton-iteráció minden lépésében szükséges a derivált adott pontbeli értéke.

Ha a derivált számítása nem lehetséges, vagy túl költséges, akkor az $f'(x_k) \approx [x_{k-1}, x_k]f$ közelítést alkalmazhatjuk.

$$x_{k+1} = x_k - \frac{f(x_k)}{[x_{k-1}, x_k]f} = x_k - f(x_k)\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}$$

Ez a **szelőmódszer**.

Szelőmódszer

x_0, x_1 kezdőpontok,

$$x_{k+1} = x_k - f(x_k)\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}, \quad k = 1, 2, \ldots$$

A képlet hasonló a húrmódszerhez, de itt nem vizsgáljuk az új pontban a függvény előjelét, mindig a 2 utolsó pontból számítjuk a következőt.

- a képlet nem feltétlenül definiált ($f(x_k) = f(x_{k-1})$ lehet)
- 2 pontra támaszkodó

Szelőmódszer

Konvergencia feltételei ugyanazok, mint a Newton-iterációnál, csak még $|x_1 - x^*| < \frac{2m_1}{M_2}$ is kell.

A konvergenciarend alacsonyabb, mint a Newton-iterációnál:

$$|x_{k+1} - x^*| \le C|x_k - x^*|^p,$$

ahol $p = \frac{1+\sqrt{5}}{2} \approx 1.618$.

(Húrmódszernél $p = 1$, Newton-módszernél $p = 2$.)

Nemlineáris egyenletek megoldása Matlab-bal

Például az `fzero` vagy `fsolve` függvényt használhatjuk.

```
[xopt,fopt] = fzero(F,x0)
```

vagy

```
[xopt,fopt] = fsolve(F,x0)
```

Az $F(x) = 0$ egyenlet gyökét közelíti az `x0` kezdőértékből indulva. Az `fsolve` esetén F akár egy többváltozós függvény is lehet.

`xopt` : a gyök közelítése,
`fopt` : a függvény értéke `xopt`-ban

Az `fzero` csak olyan gyököt tud megtalálni, ahol előjelet vált a függvény, egyébként használjuk az `fminbnd` Matlab-függvényt.

Példa

Közelítsük a $\cos(x) = 3x$ egyenlet megoldását az `fsolve` segítségével.

Írjuk át az egyenletet $F(x) = 0$ alakba:

$$\cos(x) - 3x = 0.$$

Hívjuk meg az `fsolve` függvényt pl $x_0 = 0$ kezdőértékkel.

```
>> F=@(x) cos(x)-3*x;
>> [xopt,fopt]=fsolve(F,0)
xopt =  0.31675
fopt =   -2.8390e-09
```

Példa

Matlab segítségével határozzuk meg az $\cos(4x^3) - x^2 \sin(x) = 0$ egyenlet 0-hoz legközelebbi gyökét.

Próbálkozzunk előbb az `fsolve`-val!

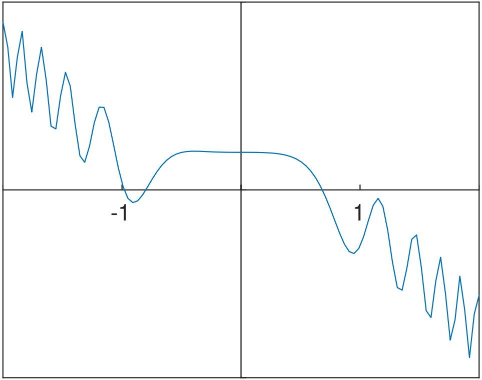
```
>>f=@(x) cos(4*x.^3)-x.^2.*sin(x);
>>[xopt,fopt]=fsolve(f,0)

No solution found.
fsolve stopped because the problem appears regular as
measured by the gradient, but the vector of function
values is not near zero as measured by the
value of the function tolerance.
<stopping criteria details>
xopt =
     0
fopt =
     1
```

Próbálkozzunk az `fzero` függvénnyel is!

```
>> [xopt,fopt]=fzero(f,0)

xopt =

   -0.8000

fopt =

   1.6653e-16
```

Azt kaptuk, hogy az $x = -0.8$ a függvény egy zérushelye.

Ábrázoljuk a függvényt a $[-2, 2]$ intervallumon!



Az ábra alapján úgy tűnik nem a 0-hoz legközelebbi gyököt találtuk meg. Próbálkozzunk más kezdőértékkel!

```
>> [xopt,fopt]=fsolve(f,0.5)

Equation solved.

fsolve completed because the vector of function values is
near zero as measured by the value of the function tolerance,
and the problem appears regular as measured by the gradient.

<stopping criteria details>

xopt =

    0.6826

fopt =

  -4.1100e-10
```

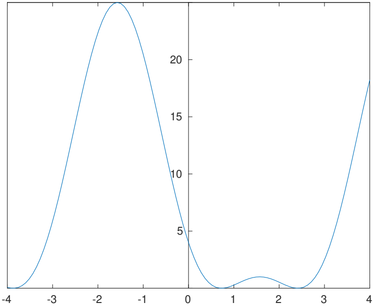
```
>> [xopt,fopt]=fzero(f,0.5)

xopt =

    0.6826


fopt =

   2.2204e-16
```

Mindkét függvény ugyanazt a gyököt találta meg.

fminbnd

- `x=fminbnd(f,xmin,xmax)`
- `[x,fval,exitflag,output]=fminbnd(f,xmin,xmax)`

Megkeresi az egyváltozós `f` függvény $[xmin, xmax]$ intervallumbeli minimumát. Alkalmas olyan gyökök megkeresésére, ahol nem vált előjelet a függvény.



Az $f(x) = 13 - 9\cos^2(x) - 12\sin(x)$ függvény.

Példa

Keressük meg az $f(x) = 13 - 9\cos^2(x) - 12\sin(x)$ függvény $[-4, 4]$-beli minimumhelyeit.

```
>> f=@(x) 13-9*cos(x).^2-12*sin(x);
>> [x,fval]=fminbnd(f,0,1)
x =
    0.7297
fval =
   9.2491e-11
>> [x,fval]=fminbnd(f,2,3)
x =
    2.4119
fval =
   1.7231e-13
```

Nemlineáris egyenletrendszerek.

$f(x) = 0$, ahol $f : \mathbb{R}^n \to \mathbb{R}^n$. Másképpen:

$$\begin{aligned} f_1(x_1, \dots, x_n) &= 0 \\ f_2(x_1, \dots, x_n) &= 0 \\ &\vdots \\ f_n(x_1, \dots, x_n) &= 0 \end{aligned}$$

Példa: $f : \mathbb{R}^2 \to \mathbb{R}^2$,

$$\begin{aligned} -4x_1 + \cos(2x_1 - x_2) - 3 &= 0 \\ -3x_2 + \sin x_1 - 2 &= 0 \end{aligned}$$

Példa

Közelítsük az előző egyenletrendszer megoldását az `fsolve` segítségével.

```
>> F=@(x) [-4*x(1)+cos(2*x(1)-x(2))-3; -3*x(2)+sin(x(1))-2];
>> [xopt,fopt]=fsolve(F,[0;0])
xopt =

  -0.50406
  -0.82766

fopt =

  -3.9646e-09
   3.1555e-10
```

Polinomok gyökei

Polinomok gyökeinek közelítésére a `roots` függvényt használhatjuk:

```
r=roots(p)
```

ahol a `p` vektorban a polinom együtthatóit kell felsorolni a főegyütthatóval kezdve.

Példa

Közelítsük a $p(x) = 2x^3 - x + 1$ polinom gyökeit.

```
>> p=[2 0 -1 1]
>> r=roots(p)
r =

  -1.00000 + 0.00000i
   0.50000 + 0.50000i
   0.50000 - 0.50000i
```

5. feladat

(a) Közelítse a $3x = \cos(x)$ egyenlet gyökeit!

(b) Közelítse a $3x^3 - 12x + 4 = 0$ egyenlet gyökeit!

(c) Közelítse az $e^x = \sin(x)$ egyenlet gyökeit!

(d) Közelítse az $\ln(x) = 2 - x$ egyenlet gyökét!

(e) Közelítse a $\cos^2(x) + 2\sin(x) = 2$ egyenlet gyökét!

(f) Közelítse az $x^4 - x^3 - 2x^2 - 2x + 4 = 0$ egyenlet gyökeit!

Egyváltozós függvény szélsőértéke (emlékeztető)

Az $f : \mathbb{R} \to \mathbb{R}$ függvény szélsőértékhelyeit keressük.

Szükséges feltétel

Az $f : \mathbb{R} \to \mathbb{R}$ differenciálható függvénynek csak ott lehet lokális szélsőértéke, ahol $f'(x) = 0$.

Elégséges feltételek

Ha az $f : \mathbb{R} \to \mathbb{R}$ függvény kétszer differenciálható és

- $f'(x^\star) = 0$, $f''(x^\star) > 0$, akkor f-nek $x^\star$-ban lokális minimuma van.
- $f'(x^\star) = 0$, $f''(x^\star) < 0$, akkor f-nek $x^\star$-ban lokális maximuma van.

Példa

Egy légitársaság A és B város közötti repülőjáratára 500 Euró egy jegy. A két város között egy 300 férőhelyes gép közlekedik, de átlagosan csak 180 utassal. Piackutatások szerint minden 5 Eurós engedmény a jegyárból átlagosan 3 plusz utast jelentene. Milyen jegyár mellett lenne maximális a légitársaság bevétele?

Tegyük fel, hogy a légitársaság $5n$ Eurót enged a jegyárból. Ekkor a várható bevétele:

$$f(n) = (180 + 3n)(500 - 5n) = -15n^2 + 600n + 90000$$

Az f maximumhelyét keressük.

$$f'(n) = -30n + 600$$

$$f'(n) = 0 \iff n = 20$$

Mivel

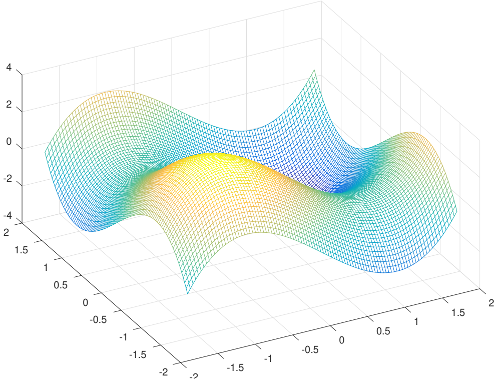
$$f''(n) = -30,$$

így $f''(20) < 0$, azaz $n = 20$ az f függvény maximumhelye.

Többváltozós függvények minimalizálása

Egy $f : \mathbb{R}^n \to \mathbb{R}$ függvény (lokális) minimumhelyét keressük.

Például: $f : \mathbb{R}^2 \to \mathbb{R}$, $f(x) = x_1^3 + x_2^3 - 3x_1 - 3x_2$.



Rajzoltassuk ki a függvényt!

```
f=@(x,y) x.^3-3*x+y.^3-3*y;
figure; fmesh(f,[-2,2])
```

Másik lehetőség:

```
xx=linspace(-2,2);
yy=xx;
[X,Y]=meshgrid(xx,yy);
Z=X.^3+Y.^3-3*X-3*Y;
figure; mesh(X,Y,Z)
```

Ilyenkor az `X` és `Y` mátrixok tartalmazzák az `xx` és `yy` vektorokban adott pontok által meghatározott rácspontok x és y koordinátáit. (Az `fmesh` esetén ezt automatikusan számítja a Matlab.)
Mindkét esetben a függvény értékét egyszerre számítjuk az összes rácspontban, ne feledkezzünk el a pontonkénti műveletekről!

Gradiens

Az $f : \mathbb{R}^n \to \mathbb{R}$ függvény x-beli gradiense

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(x) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x) \end{pmatrix}$$

Példa

Az $f(x) = x_1^3 + x_2^3 - 3x_1 - 3x_2$ függvény gradiense:

$$\nabla f(x) = \begin{pmatrix} 3x_1^2 - 3 \\ 3x_2^2 - 3 \end{pmatrix}$$

A lokális szélsőérték feltételei

Elsőrendű szükséges feltétel

Ha x^* az $f : \mathbb{R}^2 \to \mathbb{R}$ lokális minimumhelye, és f folytonosan differenciálható az x^* egy nyílt környezetében, akkor $\nabla f(x^*) = 0$.

Definíció (Stacionárius pont)

Legyen $f : \mathbb{R}^2 \to \mathbb{R}$. Az x^* pontot stacionárius pontnak hívjuk, ha $\nabla f(x^*) = 0$.

Megjegyzés

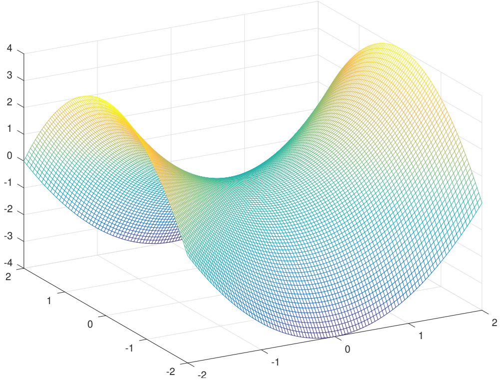
Ha x^* stacionárius pontja f-nek, akkor stacionárius pontja $-f$-nek is, azaz a stacionárius pont lokális maximum is lehet.

Definíció (Nyeregpont)

Ha x^* olyan stacionárius pontja f-nek, amely se nem lokális minimum, se nem lokális maximum, akkor nyeregpontnak hívjuk.

Példa

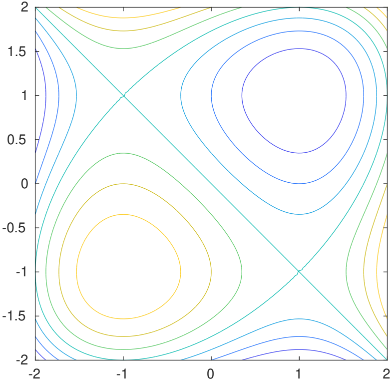
Legyen $f(x) = x_1^2 - x_2^2$. Ekkor $\nabla f(x) = (2x_1, -2x_2)^T$, így $x = (0, 0)$ az egyetlen stacionárius pont, amely nyeregpont.



Rajzoljuk ki az $f(x) = x_1^3 + x_2^3 - 3x_1 - 3x_2$ függvény szintvonalait is!

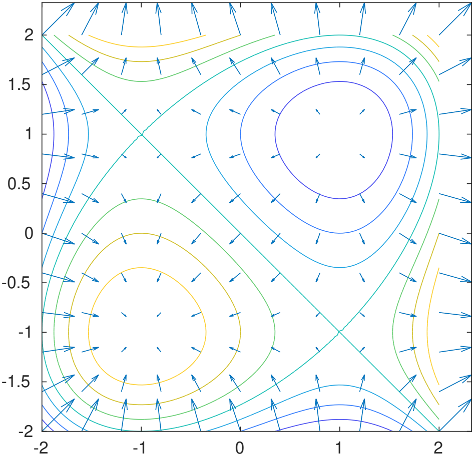
```
f=@(x,y) x.^3-3*x+y.^3-3*y;
figure; fcontour(f,[-2,2])
axis equal
```

```
xx=linspace(-2,2);
yy=xx;
[X,Y]=meshgrid(xx,yy);
Z=X.^3+Y.^3-3*X-3*Y;
figure; contour(X,Y,Z)
axis equal
```



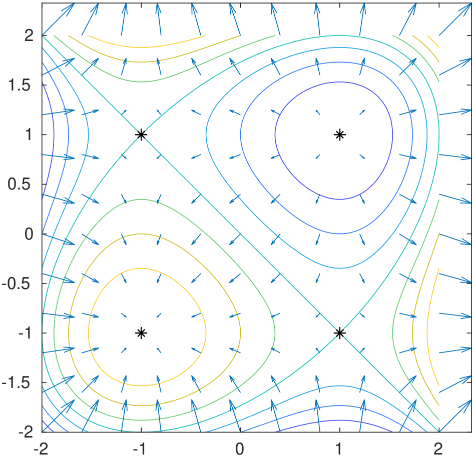
Láttuk, hogy a gradiensvektor értéke pontonként más-más lehet. Rácsozzuk be a $[-2,2]^2$ tartományt (mindkét tengely mentén 11-11 részre osztva) és számítsuk ki az előző függvény gradiensét ezekben a pontokban, majd rajzoltassuk rá ezeket a vektorokat a szintvonalakra! Használhatjuk a Matlab `gradient` függvényét is. (Ekkor nem kell kiszámolnunk a gradienst, a Matlab numerikusan közelíti azt)

```
%a szintvonalak
xx=linspace(-2,2); yy=xx;
[X,Y]=meshgrid(xx,yy);
Z=X.^3+Y.^3-3*X-3*Y;
figure; contour(X,Y,Z)
axis equal

%a gradiensmezo
xx=linspace(-2,2,11); yy=xx;
[X,Y]=meshgrid(xx,yy);
Z=X.^3+Y.^3-3*X-3*Y;
[dX,dY]=gradient(Z);
hold on; quiver(X,Y,dX,dY)
```



Az $f(x) = x_1^3 + x_2^3 - 3x_1 - 3x_2$ függvény szintvonalai és a gradiensmező.

Határozzuk meg, és tegyük rá az ábrára a stacionárius pontokat is!

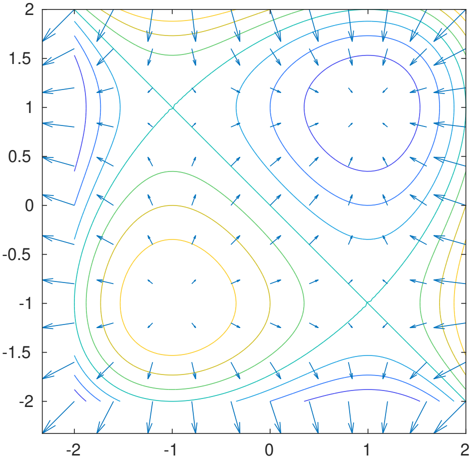


Az előző ábrán megfigyelhetjük, hogy

- a gradiensvektor merőleges az adott pontbeli szintvonalra
- a vektorok hossza a gradiens nagyságát, az iránya a gradiens irányát mutatja
- bizonyos pontokban a gradiensvektor hossza 0, vagy 0 közeli

A gradiensvektor az adott pontban a legmeredekebb emelkedés irányába mutat, a (-1)-szerese (a negatív gradiens) pedig a legmeredekebb csökkenés irányába.

Ha a gradiensmező helyett a negatív gradiensmezőt rajzoltatjuk ki, akkor a nyilak a csökkenés irányába mutatnak.
Ekkor az előző kód utolsó utasítása: `quiver(X,Y,-dX,-dY)`



Az $f(x) = x_1^3 + x_2^3 - 3x_1 - 3x_2$ függvény szintvonalai és a **negatív** gradiensmező.

Többváltozós függvények minimalizálása Matlab-bal

Az `fminsearch` vagy az `fminunc` függvényeket használhatjuk.

Példa: Keressük meg az előző függvény egy lokális minimumhelyét.

Mindkét Matlab-függvény hívásához meg kell adnunk a minimumhely egy kezdeti közelítését.

```
>> f=@(x) x(1)^3+x(2)^3-3*x(1)-3*x(2);
>> [xopt,fopt]=fminsearch(f,[0.5,0.5])

xopt =
    1.0000    1.0000

fopt =
   -4.0000
```

Az `fminunc` függvénnyel:

```
>> f=@(x) x(1)^3+x(2)^3-3*x(1)-3*x(2);
>> [xopt,fopt]=fminunc(f,[0.5,0.5])

Local minimum found.

Optimization completed because the size of the gradient is
less than the default value of the optimality tolerance.

<stopping criteria details>

xopt =
     1     1

fopt =
    -4
```

Mindkét optimalizáló függvény esetén átállíthatjuk a közelítés pontosságát, a maximális iterációszámot, ld. a függvények help-jét.

Az `fminsearch` függvény deriváltmentes módszert használ a minimumhely közelítésére

Az `fminunc` függvény használja a függvény parciális deriváltjait, illetve azoknak a numerikus közelítését.

Természetesen mindkét függvény használható egyváltozós függvények minimalizálására is. Egyváltozós függvényeket az `fminbnd` függvénnyel is minimalizálhatunk.

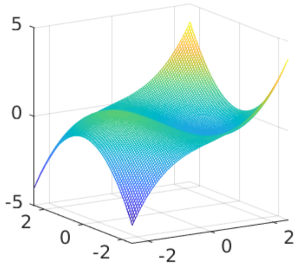
6. feladat

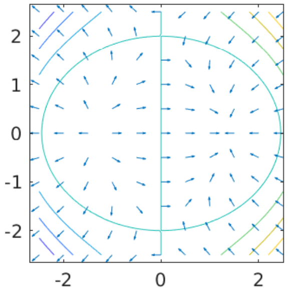
Rajzoltassa ki a megadott tartomány felett az alábbi kétváltozós függvényeket, a szintvonalaikat, a negatív gradiensmezőt és közelítse az adott tartományon belül a minimumhelyüket.

- $f(x_1, x_2) = \frac{1}{6}x_1^3 - x_1 + \frac{1}{4}x_1 x_2^2$, ha $x \in [-2.5, 2.5]^2$
- $f(x_1, x_2) = \sin(x_1)\cos(x_2)$, ha $x \in [0, 2\pi) \times [0, 2\pi)$
- $f(x_1, x_2) = x_2(1 - x_1^2 - x_2^2)$, ha $x \in [-1.5, 1.5]^2$

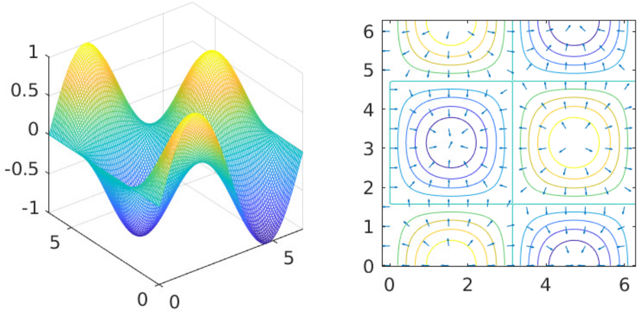
Megjegyzés: Ha a gradiensmező segítségével szeretnénk kezdőértéket találni a minimalizáló függvények számára, és a gradiensvektorok iránya azok rövidsége miatt nehezen kivehető, akkor normálhatjuk a vektorokat. Ekkor elveszítjük a hosszukkal kapcsolatos információt, de jobban látszik az irányuk. A negatív gradiensmező normált vektorokkal:

```
[dX,dY]=gradient(Z);
L=sqrt(dX.^2+dY.^2);
quiver(X,Y,-dX./L,-dY./L,0.3)
```

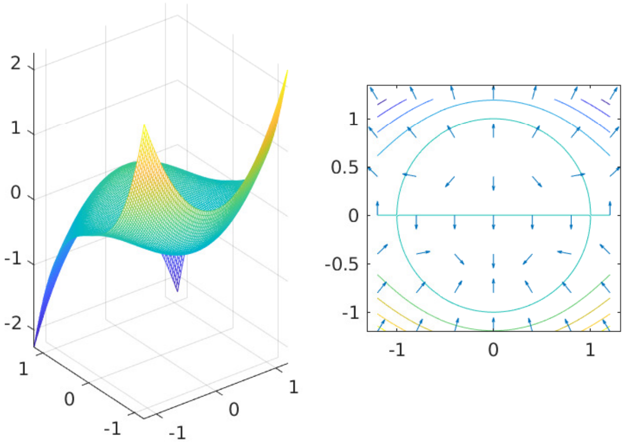




Az $f(x_1, x_2) = \frac{1}{6}x_1^3 - x_1 + \frac{1}{4}x_1x_2^2$ függvény, a szintvonalai és a normált negatív gradiens mező.



Az $f(x_1, x_2) = \sin(x_1)\cos(x_2)$ függvény, a szintvonalai és a normált negatív gradiens mező.



A $f(x_1, x_2) = x_2(1 - x_1^2 - x_2^2)$ függvény, a szintvonalai és a normált negatív gradiens mező.