Numerisk Analys, MMG410. Lecture 9.

Ickelinjära ekvationer

Betrakta ekvationen

$$f(x) = 0, \; f : \mathbb{R} \to \mathbb{R}$$

Vi kan också betrakta system av ekvationer:

$$\begin{cases} f(x, y, z) = 0 \\ g(x, y, z) = 0 \\ h(x, y, z) = 0 \end{cases}$$

Exempel

$$\begin{cases} x^2 + y^2 - 2 = 0 \\ x - y = 0 \end{cases}$$

med rötter $(1, 1)$ och $(-1, -1)$.

En ickelinjär ekvation kan ha $0, 1, 2, 3, ..., \infty$ lösningar. Ett linjärt problem $Ax = b$ kan ha $0, 1$ eller ∞ många lösningar.

Ickelinjära ekvationer

Det kan tänkas att f är definierad via en procedur, t.ex.

$$f(x) = \int_{-4}^{x} (1+t)e^{-t^2} \sin t \, dt$$

Flertalet metoder:

- Startas med en (eller flera) approximation(er)
- Skapar en sekvens av approximationer som förhoppningsvis konvergerar mot nollstället
- Kan divergera
- Försöker att hitta ett nollställe åt gången

Ickelinjära ekvationer (Bisektionsmetoden)

Givet en kontinuerlig funktion f och $p, n \in \mathbb{R}$ med $f(n) < 0$, $f(p) > 0$.

```
while |n - p| > tol do
         m = (n + p)/2
         if f(m) < 0 then ! Ta hand om exakt likhet?
                 n = m
         else
                 p = m
        endif
end
```

Om begynnelseintervallet har längden τ har intervallet längden

$$\frac{\tau}{2^k}$$

efter k iterationer.

Ickelinjära ekvationer (Bisektionsmetoden)

Bisektionsmetodens fördelar

- räcker att f är kontinuerlig
- konvergerar alltid för f kontinuerlig
- får ett intervall där roten ligger
- deterministiskt i antal steg

och nackdelar

- kan ej generaliseras till system
- långsam
- kan vara svårt att hitta p och n

"långsam men säker"

Snabbare metoder: lös ett svårt problem genom att lösa en sekvens av enklare problem.

Ickelinjära ekvationer (Sekantmetoden)

Linjärisering, approximera f med en linjär funktion. Sekanten (den räta linjen) har ekvationen

$$y(x) = \frac{f(b) - f(a)}{b - a}(x - a) + f(a)$$

varför den nya approximationen c ges utav

$$c = a - f(a)\frac{b - a}{f(b) - f(a)} = \frac{af(b) - bf(a)}{f(b) - f(a)}$$

Iterera: givet två startvärden x_0, x_1

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

Om f är linjär ger denna algoritm nollstället i ett steg.

Matlabs program (halveringsmetoden och sekantmetoden) för $f(x) = x^3 - 2x - 5 = 0$

```
close all; tolerance = 10e-15;
fun = @(x)x^3 - 3*x - 5;
% definition av intervalet [n,p]
n= 0; p=3; it=0;
% Halveringsmetoden
while abs(n - p) > tolerance
m = (n + p)/2;
if fun(m)< 0
n = m;
else
p = m;
end
it = it+1;
func_val(it) =fun(p); iteration(it) = it;
end
```

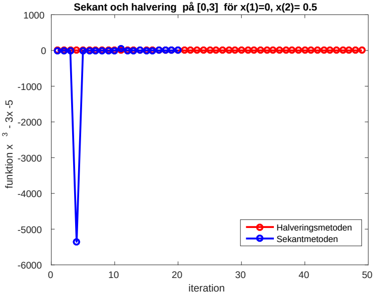
Matlabs program (fortsättningen)

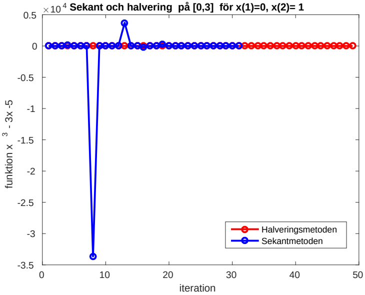
```
%Sekantmetoden
x=0; iteration_sekant = 0;
func_sekant = 0;
k=2;
%Initialiseringen
x(1) = 2.0; x(2) = 1.0;
iteration_sekant(1) = 1; iteration_sekant(2) = 2;
func_sekant(1) =fun(x(1)); func_sekant(2) =fun(x(2));
while abs( x(k) - x(k-1) ) > tolerance
numerator = fun(x(k))*(x(k-1) - x(k));
denominator = fun(x(k-1)) - fun(x(k));
x(k+1) = x(k) - numerator/denominator;
k = k+1
iteration_sekant(k) = k;
func_sekant(k) =fun(x(k));
end
```

Matlabs program (fortsättningen)

```
figure
plot(iteration, func_val,'o  r-', 'LineWidth',2)
hold on plot(iteration_sekant,func_sekant, 'o  b-', 'LineWidth',2)
xlabel('iteration')
ylabel('funktion x^3 - 3x - 5')
legend('Halveringsmetoden','Sekantmetoden')
title(['Sekant och halvering på [0,3] för x(1)=',num2str(x(1)),', x(2)=
',num2str(x(2))])
```

Example: halveringsmetoden och sekantmetoden

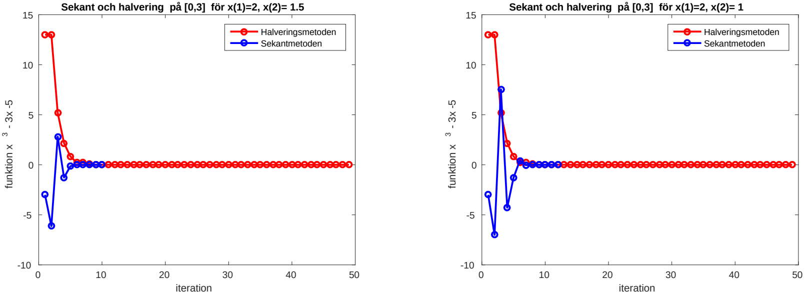




a) startvärde $x(1) = 0, x(2) = 0.5$ b) startvärde $x(1) = 0, x(2) = 1$

tolerance = 10e-15; Beräknad x i halveringsmetoden:
x = 2.2790
Beräknad x i sekantmetoden:
x = 2.2790
nr.iterations i halveringsmetoden: a), b) 49, nr.iterations i sekantmetoden: a) 20, b) 31;

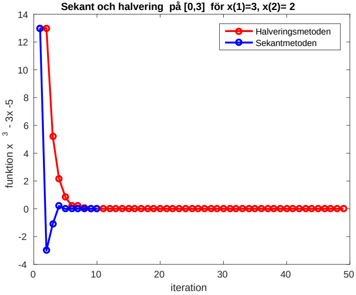
Example: halveringsmetoden och sekantmetoden

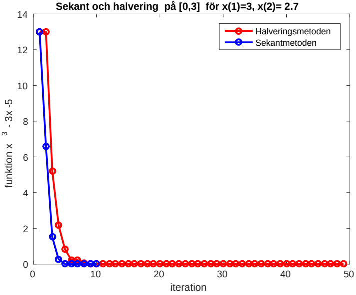


a) startvärde $x(1) = 2, x(2) = 1.5$ b) startvärde $x(1) = 2, x(2) = 1$

tolerance = 10e-15; Beräknad x i halveringsmetoden:
x = 2.2790
Beräknad x i sekantmetoden:
x = 2.2790
nr.iterations i halveringsmetoden: a), b) 49, nr.iterations i sekantmetoden: a) 10, b) 12;

Example: halveringsmetoden och sekantmetoden





a) startvärde $x(1) = 3, x(2) = 2$ b) startvärde $x(1) = 3, x(2) = 2.7$

tolerance = 10e-15; Beräknad x i halveringsmetoden:
x = 2.2790
Beräknad x i sekantmetoden:
x = 2.2790
nr.iterations i halveringsmetoden: a), b) 49, nr.iterations i sekantmetoden: a), b) 10.

Ickelinjära ekvationer (Newtons metod)

Vi vill lösa ekvation $f(x) = 0$.
Taylor's theorem:

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

för $Q \in [x, x+h]$, eller vi kan skriva Taylor's theorem som:

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(Q)(x - x_0)^2}{2!},$$

för $Q \in [x_0, x], h = x - x_0$.

$$0 = f(x) \approx f(x_0) + f'(x_0)(x - x_0),$$

som vi skriver om:

$$f(x_0) + f'(x_0)(x - x_0) \approx 0$$

Vi kan beräkna x frn ovanstpende ekvation som:

$$x - x_0 \approx -\frac{f(x_0)}{f'(x_0)}, \ x \approx x_0 - \frac{f(x_0)}{f'(x_0)}.$$

Ickelinjära ekvationer (Newtons metod)

Kan approximera med tangenten istället för sekanten (Newton-Raphson, 1690). Tangenten har ekvationen:

$$y = f(a) + f'(a)(x - a)$$

När $x = c$ (en rot) är $y = 0$. Alltså

$$c = a - \frac{f(a)}{f'(a)}$$

Iterera:

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

Kräver endast <u>ett</u> startvärde, men måste ha <u>derivatan</u>. Newtons eget exempel: $x^3 - 2x - 5 = 0$. Iterationen blir

$$x_{k+1} = x_k - \frac{x_k^3 - 2x_k - 5}{3x_k^2 - 2}$$

Algorithm: Newtons's metod för 1-D ickelinjär ekvation

- x_0 = initial guess;
- for $k = 0, 1, 2, ...$
  $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$
  end

Matlabs program (halveringsmetoden, sekantmetoden och Newton's metod) för $f(x) = x^3 - 2x - 5 = 0$

```
close all; tolerance = 10e-10;
fun = @(x)x^3 - 3 * x - 5;
% definition av intervalet [n,p]
n= 0; p=3; it=0;
% Halveringsmetoden
while abs(n - p) > tolerance
m = (n + p)/2;
if fun(m)< 0
n = m;
else
p = m;
end
it = it+1;
func_val(it) =fun(p); iteration(it) = it;
end
```

```
%Sekantmetoden
x=0; iteration_sekant = 0;
func_sekant = 0;
k=2;
%Initialiseringen
x(1) = 2.0; x(2) = 1.0;
iteration_sekant(1) = 1; iteration_sekant(2) = 2;
func_sekant(1) =fun(x(1)); func_sekant(2) =fun(x(2));
while abs( x(k) - x(k-1) ) > tolerance
numerator = fun(x(k))*(x(k-1) - x(k));
denominator = fun(x(k-1)) - fun(x(k));
x(k+1) = x(k) - numerator/denominator;
k = k+1
iteration_sekant(k) = k;
func_sekant(k) =fun(x(k));
end
```

Matlabs program (fortsättningen)

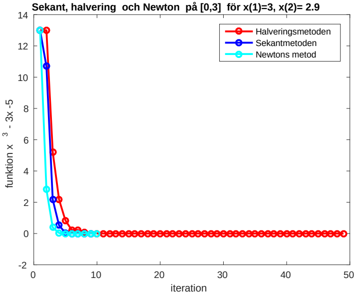
```
figure
plot(iteration, func_val,'o  r-', 'LineWidth',2)
hold on
plot(iteration_sekant,func_sekant, 'o  b-', 'LineWidth',2)
% Newtons metod
y=0; iteration_newton = 0; func_newton = 0;
%Initialiseringen
y(1) = 3.0; k=2;
iteration_newton(1) = 1; func_newton(1) =fun(y(1));
numerator = fun(y(1)); denominator = 3.0*y(1)^2 − 2.0;
y(2) = y(1) - numerator/denominator;
% Main Newton's iterations
while abs( y(k) - y(k-1) )> tolerance
numerator = fun(y(k)); denominator = 3.0*y(k)^2 − 2.0;
y(k+1) = y(k) - numerator/denominator;
iteration_newton(k) = k; func_newton(k) =fun(y(k));
k = k+1;
end
```

Matlabs program (fortsättningen)

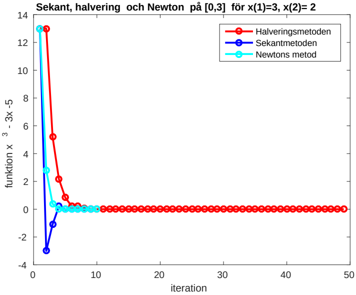
plot(iteration_newton,func_newton, 'o c-','LineWidth',2)
xlabel('iteration')
ylabel('funktion $x^3 - 3x - 5'$)
legend('Halveringsmetoden','Sekantmetoden','Newtons metod')
title(['Sekant, halvering och Newton på [0,3] för x(1)=',num2str(x(1)),',
x(2)= ',num2str(x(2))])

Example: halveringsmetoden, sekantmetoden och Newton's metod för $f(x) = x^3 - 2x - 5 = 0$

Startvärde i Newton's metod: $x(1) = 3.0$, tolerance = 10e-10.



a) startvärde i sekantmetod: $x(1) = 3.0, x(2) = 2.9$

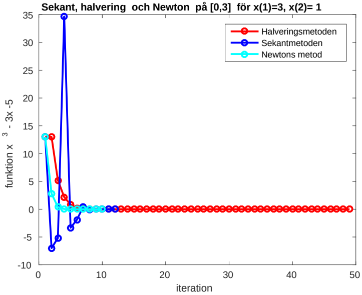


b) startvärde i sekantmetod: $x(1) = 3, x(2) = 2.0$

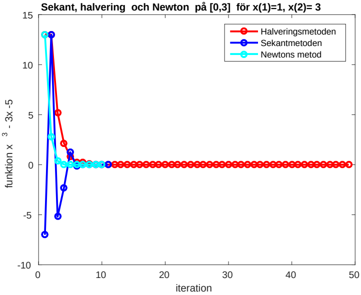
Beräknad x i halveringsmetoden, sekantmetoden och Newton's metoden:
x = 2.2790
nr.iterations i halveringsmetoden: a), b) 49, nr.iterations i sekantmetoden: a), b) 9; i Newton's metod : a), b) 10.

Example: halveringsmetoden, sekantmetoden och Newton's metod för $f(x) = x^3 - 2x - 5 = 0$

Startvärde i Newton's metod: $x(1) = 3.0$, tolerance = 10e-10.



a) startvärde i sekantmetod:
$x(1) = 3.0, x(2) = 1.0$



b) startvärde i sekantmetod:
$x(1) = 1, x(2) = 3.0$

Beräknad x i halveringsmetoden, sekantmetoden och Newton's metoden:
x = 2.2790
nr.iterations i halveringsmetoden: a), b) 49, nr.iterations i sekantmetoden: a), 12 b) 11; i Newton's metod : a), b) 10.

Repetition av Taylors formel.

$$\begin{bmatrix} f(a+h,b+k) \\ g(a+h,b+k) \end{bmatrix} = \begin{bmatrix} f(a,b) \\ g(a,b) \end{bmatrix} + \begin{bmatrix} \frac{\partial f(a,b)}{\partial x} h & \frac{\partial f(a,b)}{\partial y} k \\ \frac{\partial g(a,b)}{\partial x} h & \frac{\partial g(a,b)}{\partial y} k \end{bmatrix} + \ldots =$$

$$\begin{bmatrix} f(a,b) \\ g(a,b) \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{\partial f(a,b)}{\partial x} & \frac{\partial f(a,b)}{\partial y} \\ \frac{\partial g(a,b)}{\partial x} & \frac{\partial g(a,b)}{\partial y} \end{bmatrix}}_{J(a,b)} \begin{bmatrix} h \\ k \end{bmatrix} + \ldots$$

Matrisen av partiella derivator, $J(a,b)$, kallas Jacobianen.

Vi står i (x_j, y_j) och vill hitta korrektioner, (h,k), så att $f(x_j+h, y_j+k) = 0$ och $g(x_j+h, y_j+k) = 0$.

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} f(x_j+h, y_j+k) \\ g(x_j+h, y_j+k) \end{bmatrix} \approx \begin{bmatrix} f(x_j, y_j) \\ g(x_j, y_j) \end{bmatrix} + J(x_j, y_j) \begin{bmatrix} h \\ k \end{bmatrix}$$

Om Jacobianen J är ickesingulär kan vi få de approximativa korrektionerna:

Ickelinjära ekvationer (Newton för system)

$$\begin{bmatrix} h \\ k \end{bmatrix} \approx -J^{-1}(x_j, y_j) \begin{bmatrix} f(x_j, y_j) \\ g(x_j, y_j) \end{bmatrix}$$

Iterera!

$$\begin{bmatrix} x_{j+1} \\ y_{j+1} \end{bmatrix} = \begin{bmatrix} x_j \\ y_j \end{bmatrix} - J^{-1}(x_j, y_j) \begin{bmatrix} f(x_j, y_j) \\ g(x_j, y_j) \end{bmatrix}$$

Jämför med det skalära fallet:

$$x_{j+1} = x_j - f(x_j)/f'(x_j)$$

Vi räknar naturligtvis inte ut inversen utan löser systemet:

$$J(x_j, y_j)\, c = \begin{bmatrix} f(x_j, y_j) \\ g(x_j, y_j) \end{bmatrix}, \text{ med } c = -\begin{bmatrix} h \\ k \end{bmatrix}$$

Ickelinjära ekvationer (Newton för system)

Betrakta som exempel ekvationen

$$\begin{cases} x^2 + y^2 = 2 \\ xy = \frac{1}{2} \end{cases}$$

Vi skriver problemet på normalform (nollor i ena ledet), så att våra funktioner blir

$$\begin{cases} f(x,y) = x^2 + y^2 - 2 \\ g(x,y) = xy - \frac{1}{2} \end{cases}$$

Newtons metod blir då

$$\begin{bmatrix} x_{j+1} \\ y_{j+1} \end{bmatrix} = \begin{bmatrix} x_j \\ y_j \end{bmatrix} - \begin{bmatrix} 2x_j & 2y_j \\ y_j & x_j \end{bmatrix}^{-1} \begin{bmatrix} x_j^2 + y_j^2 - 2 \\ x_j y_j - \frac{1}{2} \end{bmatrix}$$

Om vi startar i $x_0 = -3$ och $y_0 = 10$ får vi följande approximationer:

```
-3.0000e+00  -1.4121e+00  -5.4236e-01 -1.4188e-02
-1.0000e+01   5.1264e+00   2.8033e+00  1.8081e+00
```

Ickelinjära ekvationer (Newton för system)

```
 2.7380e-01  3.5877e-01  3.6597e-01  3.6603e-01
 1.4593e+00  1.3733e+00  1.3661e+00  1.3660e+00

 3.6603e-01  3.6603e-01  3.6603e-01
 1.3660e+00  1.3660e+00  1.3660e+00

>> fel =
-3.3660e+00 -1.7781e+00 -9.0838e-01 -3.8021e-01
 8.6340e+00  3.760e+00  1.4373e+00  4.4208e-01

-9.2230e-02 -7.2583e-03 -5.1931e-05 -2.6966e-09
-9.3297e-02  7.2586e-03  5.1931e-05  2.6966e-09

          0           0           0
          0           0           0
```

Om man arbetar med stora system kan man inte ha variabler för $x, y, z, w, \ldots$ utan får använda vektorer, analogt för funktionerna.

Exemplet kan skrivas på följande vis. x och y får vara elementen x_1 resp. x_2 i vektorn x.

$$\begin{cases} x_1^2 + x_2^2 - 2 = 0 \\ x_1 x_2 - \frac{1}{2} = 0 \end{cases}$$

Vår funktion f med två komponenter blir

$$\begin{cases} f_1(x_1, x_2) = x_1^2 - x_2^2 - 2 \\ f_2(x_1, x_2) = x_1 x_2 - \frac{1}{2} \end{cases}$$

Normalt skriver vi bara $f(x) = 0$ där f, x och 0 är vektorer. f är alltså en vektorvärd funktion som beror av en vektor. Newtons metod blir (notera placeringen av iterationsindex)

$$\begin{bmatrix} x_1^{(j+1)} \\ x_2^{(j+1)} \end{bmatrix} = \begin{bmatrix} x_1^{(j)} \\ x_2^{(j)} \end{bmatrix} - \begin{bmatrix} 2x_1^{(j)} & 2x_2^{(j)} \\ x_2^{(j)} & x_1^{(j)} \end{bmatrix}^{-1} \begin{bmatrix} (x_1^{(j)})^2 + (x_2^{(j)})^2 - 2 \\ x_1^{(j)} x_2^{(j)} - \frac{1}{2} \end{bmatrix}$$

Allmänt:

$$x^{(j+1)} = x^{(j)} - J^{-1}(x^{(j)}) f(x^{(j)})$$

Övningar

Sätt upp Newtons metod för följande problem:

1. $x^3 - 2x - 5 = 0$
2. $e^{-x} = x$
3. $x \sin(x) = 1$

Svar

Newtons metod:

1. $x_{k+1} = x_k - \frac{x_k^3 - 2x_k - 5}{3x_k^2 - 2}$
2. $x_{k+1} = x_k - \frac{e^{-x_k} - x_k}{-e^{-x_k} - 1}$
3. $x_{k+1} = x_k - \frac{x_k \sin(x_k) - 1}{x_k \sin(x_k) + x_k \cos(x_k)}$