Numeriska metoder, grundkurs II
Övning 1 för I2

Övningsgrupp 1

Johannes Hjorth
hjorth@nada.kth.se
Rum 163:006, Roslagstullsbacken 35
08 - 790 69 00

Kurshemsida:
`http://www.csc.kth.se/utbildning/kth/kurser/2D1240/numi07`

Material utdelat på övningarna:

`http://www.nada.kth.se/~hjorth/teaching/numi07`

Dagens program

- Hur var det man gjorde?
  Vi repeterar matlab
- Newton-Raphsons metod
  Lösning av $f(x) = 0$
- Sekantmetoden
  När derivatan $f'(x)$ *är elak*
- Lösning av ekvationssystem
  Lika många ekvationer som obekanta
- Minsta kvadratmetoden
  Lösning av överbestämda system

Administrativt

Kom ihåg att registrera er på kursen och koppla ihop `numi07` katalogen så att bägge kommer åt den:

Kleopatra som är påloggad skriver:

```
>course join numi07
>res checkin numi07
>mkdir numi07
>cd numi07
>fs sa . caesar rlidwk
>pwd
/här/står/nu/en/lång/path/numi07
```

Caesar, hennes labkompis skriver sen i samma terminalfönster:

```
>ssh my -l caesar
>course join numi07
>res checkin numi07
>ln -s /här/står/nu/en/lång/path/numi07
>exit
```

Klart! Nu kan både Kleopatra och Caesar komma åt filerna, de skapar en katalog `lab1` och sätter igång.

Hur skriver man en funktion?

Vi vill skriva en funktion i filen funk.m som tar emot tre parametrar x, a och b samt beräknar

$$f(x) = e^{-a \cdot x^2} - \ln(x + b)$$

```
function y = funk(          )

  y =
```

Vi vill även ha en funktion för att beräkna derivatan, vilken vi lägger i filen dfunk.m

```
function yp = dfunk(          )

  yp =
```

Fyll i de tomma funktionerna ovan. När ska `.*` *användas?*

Newton-Raphsons metod

Lägg den här formeln på minnet

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

För att lösa en ekvation med Newton-Raphsons metod måste vi ha den på formen $f(x) = 0$.

Antag att vi vill beräkna en rot till

$$\frac{e^{-a \cdot x^2}}{\ln(x+b)} = 1$$

där $a = 2.533 \pm 10^{-3}$ och $b = 0.543 \pm 2 \cdot 10^{-3}$.

Hur ska vi då skriva om ekvationen?

Exempel 2.11

Vi använder oss av funk.m och dfunk.m från föregående slides då vi skriver NR.m

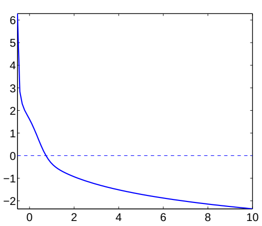
```
function x = NR(xStart, a, b, tol)

% Måste ge dx ett godtyckligt värde > tol
dx = 1; x = xStart;

% Vi använder while-loop eftersom vi inte vet hur
% många gånger loopen behöver köras

while(abs(dx) > tol)  % Kom ihåg abs
    dx = funk(x,a,b)/dfunk(x,a,b);

    % och glöm inte denna raden! ;)
    x = x - dx;

    disp([x dx])
end
```

Vilket startvärde ska vi ta?

För att hitta ett startvärde så plottar vi funktionen inom lämpligt intervall. Varför valde vi $[-0.54, 10]$?

```
clear all, close all % rensar variabler, stänger figurer
a = 2.533; da = 1e-3;
b = 0.543; db = 2e-3;

xmin = -0.54; xmax = 10;
x = linspace(xmin, xmax, 100);
plot(x, funk(x,a,b))
hold on % förhindrar att första plotten skrivs över
plot([xmin xmax], [0 0], '--')
```

Vi ser att $x_0 = 1$ verkar vara en bra startgissning



Svar med felgränser

Vi har fått osäkra indata och uppgiften säger att vi ska uppskatta felet i svaret. Vi är lata och antar att felet är linjärt runt de små störningarna.

Vi stör a uppåt, beräknar avvikelsen från det ostörda svaret, därefter stör vi b uppåt.

```
tol = 0.5e-10; x0 = 1;

x = NR(x0,a,b,tol);
xda = NR(x0,a+da,b,tol);
xdb = NR(x0,a,b+db,tol);
```

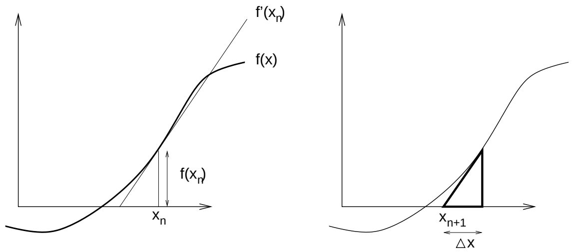
Slutligen adderar vi beloppet av de två störningarna i svaret och avrundrar uppåt!

```
xErr = abs(x - xda) + abs(x - xdb);

disp('Svar med felgräns')
disp([x xErr])
```

Matlab säger `0.7404 0.0010` och vi svarar med $x = 0.740 \pm 0.002$. Varför blev det ± 0.002?

Hur fungerar Newton-Raphson?

I varje iteration har vi ett startvärde x_n, vi känner även $f(x_n)$ samt lutning $f'(x_n)$. Rita tangenten.



Då gäller att

$$f'(x_n) = \frac{f(x_n)}{\Delta x}$$

har vi sedan Δx kan vi uppdatera x_n

$$x_{n+1} = x_n - \Delta x$$

vilket vi kan skriva om som

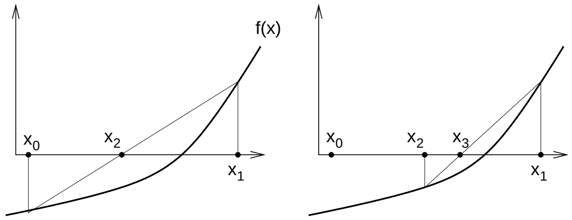
$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Sekantmetoden

Om vi inte kan beräkna derivatan kan vi istället använda sekantmetoden.

$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n)$$

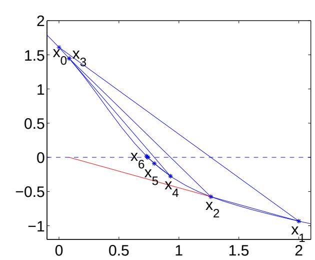
Sekantmetoden behöver *två* startgissningar och konvergerar långsammare än Newton-Raphson.



Dra en rät linje mellan de två startvärdena. Denna linjen skär x-axeln i någon punkt, som vi tar som vårt nya värde x_{n+1}.

Exempel 2.11 med sekantmetoden

```
clear all, format compact

% Startgissningar 0 och 2
x = [0 2]; tol = 1e-4; dx = 1;
a = 2.533; b = 0.543;

while(abs(dx) > tol)
    dx = (x(end) - x(end-1)) ...
        /(funk(x(end),a,b) - funk(x(end-1),a,b)) ...
        * funk(x(end),a,b);

    x(end+1) = x(end) - dx; % lägger nya värdet sist

    disp([x(end) dx])    % x(end) referar alltid till
end                      % det sista elementet i x
```



Hur löser vi ekvationssystem?

Antag att vi har fyra punkter (1,1),(2,3),(3,5),(4,-1) och vill skapa ett tredjegradspolynom som skär dem.

$$y = c_0 + c_1 \cdot x + c_2 \cdot x^2 + c_3 \cdot x^3$$

Om vi sätter in våra punkter i ekvationen ovan får vi fyra nya ekvationer

$$\begin{aligned} c_0 + c_1 \cdot 1 + c_2 \cdot 1 + c_3 \cdot 1 &= 1 \\ &= \\ &= \\ &= \end{aligned}$$

Skriv det på matrisform:

$$\begin{pmatrix} \quad & \quad & \quad & \quad \\ \\ \\ \\ \end{pmatrix} \begin{pmatrix} \quad \\ \\ \\ \\ \end{pmatrix} = \begin{pmatrix} \quad \\ \\ \\ \\ \end{pmatrix}$$

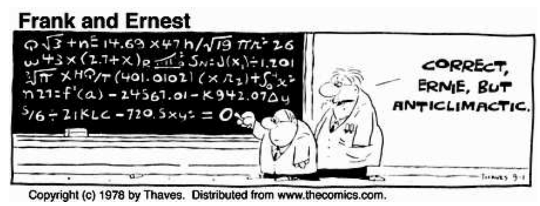
Ställ upp ekvationssystemet i matlab

Hur använder vi `x` för att bilda matrisen `A` i matlab?

Vi vill inte behöva räkna för hand eller skriva en massa ettor i matrisen.

```
x =

y =

A =




c = A\y
```

Dubbelkolla, har `x`, `y` och `A` rätt dimensioner?



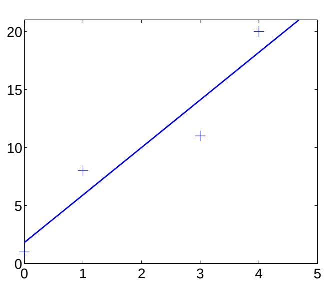
Minstakvadratmetoden – linjär algebra

Antag att vi har två punkter $(0, 1)$, $(1, 8)$. Vi kan då enkelt dra en rät linje $y = c_1 \cdot x + c_0$ genom dem.

```
A = [0 1; 1 1];
b = [1 8]';
c = A\b
```

Detta ger oss $y = 7 \cdot x + 1$.

Om vi nu får ytterligare två punkter $(3, 11)$ och $(4, 20)$ kan vi inte längre dra en rät linje som går genom alla punkterna. Hur gör vi då?



Geometrisk tolkning

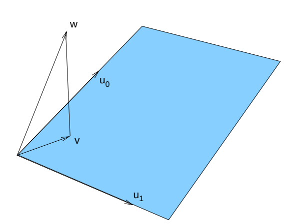
Vi har fyra ekvationer vi försöker uppfylla

$$\begin{aligned} c_0 + 0 \cdot c_1 &= 1 \\ c_0 + 1 \cdot c_1 &= 8 \\ c_0 + 3 \cdot c_1 &= 11 \\ c_0 + 4 \cdot c_1 &= 20 \end{aligned}$$

vilket kan skrivas

$$c_0 \underbrace{\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}}_{u_0} + c_1 \underbrace{\begin{pmatrix} 0 \\ 1 \\ 3 \\ 4 \end{pmatrix}}_{u_1} = \underbrace{\begin{pmatrix} 1 \\ 8 \\ 11 \\ 20 \end{pmatrix}}_{w}$$

Vektorerna u_0 och u_1 spänner upp ett plan, och vi försöker hitta den vektorn v i planet som ligger så nära w som möjligt.



Vilket fel kommer vi att göra?

Vi kommer att göra ett fel e_i vid respektive punkt.

$$e = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{pmatrix} = \underbrace{c_0 \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} + c_1 \begin{pmatrix} 0 \\ 1 \\ 3 \\ 4 \end{pmatrix}}_{v} - \underbrace{\begin{pmatrix} 1 \\ 8 \\ 11 \\ 20 \end{pmatrix}}_{w}$$

Det totala felet kan vi nu teckna på följande sätt

$$||e||_2 = \sqrt{e^{\mathsf{T}} \cdot e} = \sqrt{{e_1}^2 + {e_2}^2 + {e_3}^2 + {e_4}^2}$$

Nu vill vi bara komma på ett sätt att minimera det!

Rita ut felen e_1, e_2, e_3 och e_4 respektive felvektorn e i de två föregående figurerna!

Felvektorn e är ortogonal mot planet

Om två vektorer u och v är ortogonala blir deras skalärprodukt $u^{\mathsf{T}} \cdot v = 0$. Kan vi använda detta?

$$e = v - w$$

Om e har någon komponent i planets riktning så kan vi flytta v lite närmare w. När vi gjort det har e blivit ortogonal mot planet, vi får då att:

$$\begin{aligned} {u_1}^{\mathsf{T}} \cdot e &= 0 \\ {u_2}^{\mathsf{T}} \cdot e &= 0 \end{aligned}$$

Nu kommer vi inte närmare, e är ortogonal mot kolumnvektorerna i A. Detta kan skrivas:

$$A^{\mathsf{T}} \cdot e = 0$$

Nu blir det enklare...

Vi försöker lösa det överbestämda systemet

$$A \cdot c = w$$

Felet för vårt $c = (c_1, c_2)^{\mathsf{T}}$ kan skrivas som

$$e = A \cdot c - w$$

och det är som minst då $A^{\mathsf{T}} \cdot e = 0$.

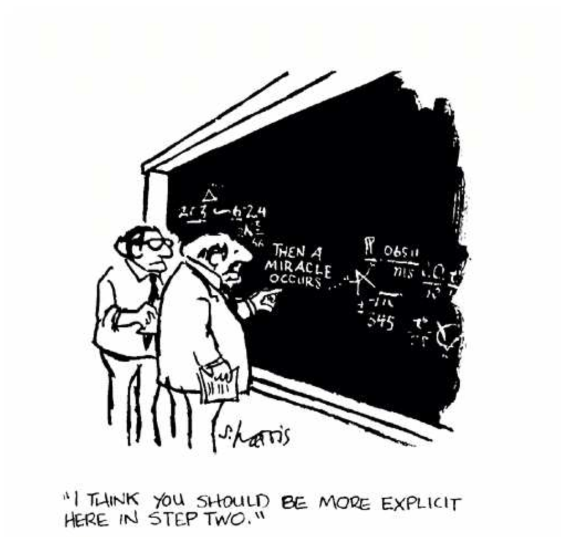
Vi skriver om det som

$$A^{\mathsf{T}} \cdot e = A^{\mathsf{T}} \cdot (A \cdot c - w) = 0$$

med andra ord

$$A^{\mathsf{T}} A \cdot c = A^{\mathsf{T}} \cdot w$$

Allt vi behöver göra är multiplicera med A^{T}



Handräkning och matlab

När vi ska lösa det överbestämda systemet för hand multiplicerar vi vänster och höger led med A^T.

$$A^\mathsf{T} A \cdot c = A^\mathsf{T} \cdot w$$

Ska vi däremot använda `matlab` ska vi *inte* multiplicera med A^T utan vi använder \-operatorn.

```
x = [0 1 3 4]';
w = [1 8 11 20]';

A = [x ones(size(x))];
c = A\w
```

Den räta linjen blir då med $c_1 = 4.1$ och $c_0 = 1.8$

$$y = c_1 \cdot x + c_0$$

Kursivt: Anledningen till att vi inte ska förmultiplicera med transponatet är därför att det försämrar ekvationens konditionstal. Matlab använder en annan matris baserad på A som inte försämrar konditionstalet lika mycket.

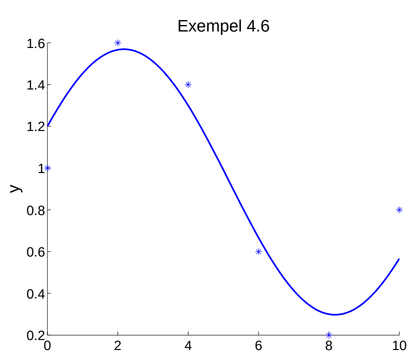
Exempel 4.6

Tidvattnet i Nordsjön bestäms av den så kallade M_2-tide, vars periodlängd är cirka tolv timmar och har formen, $H(t) = h_0 + a_1 \sin(\frac{2\pi t}{12}) + a_2 \cos(\frac{2\pi t}{12})$ där t anges i timmar. Anpassa med minstakvadratmetoden $H(t)$ till mätdataserien:

```
t = [0 2 4 6 8 10]
y = [1.0 1.6 1.4 0.6 0.2 0.8]
```

```
clear,clf
t = [0 2 4 6 8 10]';
y = [1.0 1.6 1.4 0.6 0.2 0.8]';

A = [ones(size(t)) sin(t*pi/6) cos(t*pi/6)]

AtA = A'*A     % Bara för att vi ska få se hur de ser ut
Aty = A'*y

x = A \ y      % Vi räknar direkt med A och y i matlab

title('Exempel 4.6')
xlabel('t')
ylabel('y')
hold on

plot(t, y, '*')

t2=0:0.1:10;
plot(t2, x(1) + x(2)*sin(t2*pi/6) + x(3)*cos(t2*pi/6))
```

```
>> format compact
>> exs46
A =
    1.0000         0    1.0000
    1.0000    0.8660    0.5000
    1.0000    0.8660   -0.5000
    1.0000    0.0000   -1.0000
    1.0000   -0.8660   -0.5000
    1.0000   -0.8660    0.5000
AtA =
    6.0000    0.0000   -0.0000
    0.0000    3.0000    0.0000
   -0.0000    0.0000    3.0000
Aty =
    5.6000
    1.7321
    0.8000
x =
    0.9333
    0.5774
    0.2667
```



Några bra shell kommandon

- Lista filer:
  `ls`
- Byt katalog:
  `cd numi07`
- Skapa en underkatalog:
  `mkdir lab1`
- Gå tillbaka en katalog:
  `cd ..`
- Kopiera en fil:
  `cp fil.m nyfil.m`
- Ta bort en fil:
  `rm fil.m`
- Byt namn eller flytta fil:
  `mv per.m pia.m`
- Starta matlab:
  `matlab &`

Exempel på hur det kan se ut i terminalfönstret:

```
hjorth@titanic:~$ cd numi07
hjorth@titanic:~/numi07$ ls
testfil.m
hjorth@titanic:~/numi07$ mkdir lab1
hjorth@titanic:~/numi07$ cp testfil.m lab1/nyfil.m
hjorth@titanic:~/numi07$ cd lab1
hjorth@titanic:~/numi07/lab1$ matlab &
[1] 1528
hjorth@titanic:~/numi07/lab1$ cd ..
hjorth@titanic:~/numi07$ ls
lab1  testfil.m
hjorth@titanic:~/numi07$ mv testfil.m lab1/
hjorth@titanic:~/numi07$ ls
lab1
hjorth@titanic:~/numi07$
```

Vilka filer ligger nu i `lab1` katalogen?