Newtons metod

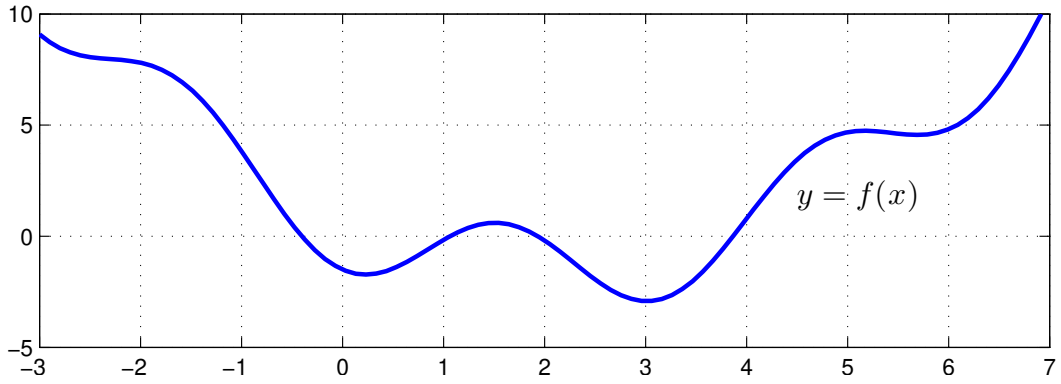
1 Inledning

Vi skall lösa ekvationer, dvs. finna nollställen till funktioner. Som exempel kan vi ta,

$$f(x) = 0.5(x-2)^2 - 2\cos(2x) - 1.5 = 0$$

Vi börjar med att rita grafen till f för att få en uppfattning om hur många nollställen vi har och ungefär var de ligger.

```
>> f=@(x)0.5*(x-2).^2-2*cos(2*x)-1.5;
>> x=linspace(-3,7);
>> plot(x,f(x))
>> axis([-3 7 -5 10]), grid on
```



Vi ser lösningar till $f(x) = 0$ som de punkter där grafen skär x-axeln. Vi kan grafiskt läsa av en första approximation av en lösning för att sedan förbättra denna med Newtons metod. Innan vi kan komma till Newtons metod måste vi dock först se på linjäriseringar av funktioner i en variabel.

2 Newtons metod

Antag att x_k är en approximation av ett nollställe till ekvationen $f(x) = 0$. Följ tangenten i punkten $(x_k, f(x_k))$, dvs.

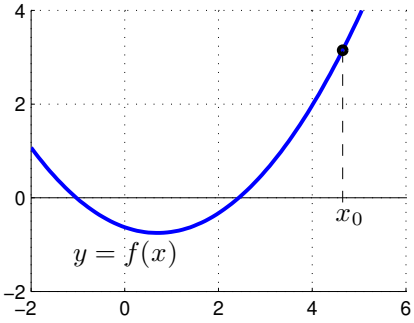
$$y = f(x_k) + f'(x_k)(x - x_k)$$

ned till x-axeln ($y = 0$) och tag skärningspunktens x-koordinat

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

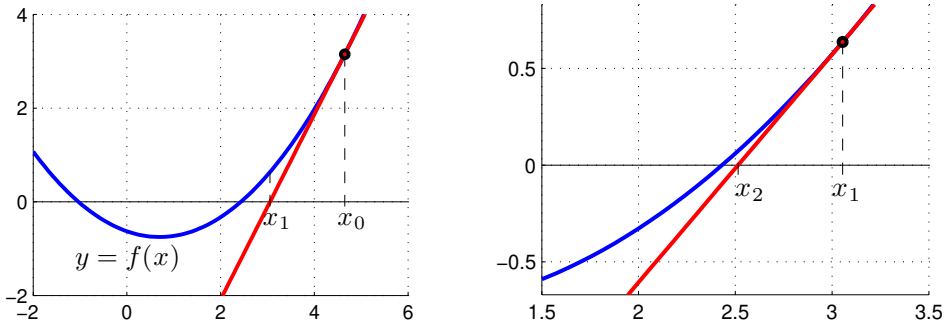
som en ny approximation av nollstället. Detta är Newtons metod.

Vi ser på några steg med metoden: Starta med en approximation x_0 av en lösning till $f(x) = 0$.



Bilda tangenten $y = f(x_0) + f'(x_0)(x - x_0)$ till f i $x = x_0$ och tag dess skärningspunkt med x-axeln som en ny approximation

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



Bilda tangenten $y = f(x_1) + f'(x_1)(x - x_1)$ i $x = x_1$ och tag dess skärningspunkt med x-axeln som en ännu nyare approximation

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

Som exempel tar vi: Lös ekvationen $f(x) = 0$ där $f(x) = \cos(x) - x$. En graf av funktionen (rita den gärna) visar att vi har ett nollställen nära $x_0 = 0.75$ som vi tar som startapproximation.

```
>> f=@(x)cos(x)-x; Df=@(x)-sin(x)-1;
>> x=0.75;
>> kmax=10; tol=0.5e-8;
>> for k=1:kmax
       h=-f(x)/Df(x);
       x=x+h;
       disp([x h])
       if abs(h)<tol, break, end
   end

   0.739111138752579  -0.010888861247421
   0.739085133364485  -0.000026005388094
   0.739085133215161  -0.000000000149324
```

I kolumnen till vänster ser vi x_k-värdena och i den till höger ser vi motsvarande $f(x_k)$-värden. Vi ser att vi får snabb konvergens, dvs. vi får snabbt ett noggrant resultat. Iterationen avbryts eftersom vi har mer är åtta korrekta decimaler (felet mindre än $\frac{1}{2} \times 10^{-8}$).

Uppgift 1. Låt $f(x) = x^3 - \cos(4x)$. Lös ekvationen $f(x) = 0$. Rita upp grafen till f för att se var ungefär lösningarna (skärningspunkterna) ligger. Hur många lösningar finns det? Läs av i grafiken en första approximation av en lösning för att sedan förbättra denna med Newtons metod. Rita ut lösningen med en liten ring. Upprepa tills du beräknat alla lösningar till ekvationen.

3 Eget program i MATLAB

Det är praktiskt att packetera en metod genom att skriva ett program eller funktion som utför metoden. Vi gjorde det i förra studioövningen för intervallhalveringsmetoden och nu gör vi det även för Newtons metod.

Uppgift 2. Skriv en function som löser ekvationen $f(x) = 0$ med Newtons metod. Funktionen skall heta `min_newton` och skall som indata ges två funktioner, dels en som beräknar $f(x)$ dels en som beräknar $f'(x)$, en startapproximation av lösningen, samt den noggrannhet lösningen skall bestämmas med. Funktionen skall som utdata ge en approximation av nollstället som uppfyller noggrannhetskravet.

Funktionen skall innehålla en hjälptext som beskriver hur den skall användas. Skriver vi `help min_newton` i Command Window så skall det se ut något liknande:

```
>> help min_newton
min_newton - beräknar nollställe till f(x) givet startapproximation x0.
  Syntax:
          x = min_newton(f,Df,x0,tol)
  Argument:
          f   - funktionshandtag: pekar på namnet till en funktionsfil eller
                till en anonym funktion. T.ex. f=@funk eller f=@(x)cos(x)-x
          Df  - funktionshandtag: pekar på namnet till en funktionsfil eller
                till en anonym funktion som ger derivatan av f.
                T.ex. Df=@Dfunk eller Df=@(x)-sin(x)-1
          x0  - ett tal som ger en startapproximation av nollstället.
          tol - positivt tal som anger önskad noggrannhet för nollstället.
  Returnerar:
          x   - ett tal som ger approximativt nollställe.
  Beskrivning:
          Programmet beräknar ett approximativt nollställe till f(x) med
  Newtons metod.
  Exempel:
          x = min_newton(@(x)cos(x)-x,@(x)-sin(x)-1,1.0,1e-5)
```

För att underlätta lite finns på studiohemsidan ett programskal `min_newton.m` att utgå ifrån.

Uppgift 3. Använd din funktion `min_newton` för att lösa följande ekvationer. Rita grafer och beräkna samtliga lösningar.

(a). $f(x) = 0.5(x-2)^2 - 2\cos(2x) - 1.5 = 0$ **(b).** $f(x) = x^3 - \cos(4x) = 0$

4 Modifiering av Newtons metod

En dålig startapproximation kan leda till att Newtons metod divergerar, då är det lämpligt att försöka med dämpad Newton

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

Dämpningsfaktorn α_k väljs så att $|f(x_{k+1})| < |f(x_k)|$. Man kan t.ex. börja med $\alpha_k = 1$, på försök ta ett steg i iterationen, om vi har fått en minskning av $|f|$ accepterar vi steget. I annat fall halverar vi successivt α_k och gör nya försök, tills vi har en minskning av $|f|$.

Om vi inte vill beräkna derivator så kan vi approximera dem med differenskvoter

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

för lämpligt valt litet positivt tal h.

5 Färdigt program i MATLAB

Verktygslådan OPTIMIZATION TOOLBOX i MATLAB har en funktion `fzero` som finner nollställen. Metoden i `fzero` är en modifiering av Newtons metod i kombination med intervallhalveringsmetoden, där den senare griper in då startapproximationen inte tillräckligt bra.

Funktion `fzero` används enligt något av alternativen

```
x=fzero(fun,x0)      x=fzero(fun,x0,opts)
```

där `fun` beskriver funktionen vi skall finna nollstället till, `x0` är en startapproximation av nollstället eller ett intervall med teckenväxling som omsluter nollstället vi söker. I senare fallet kommer `fzero` garanterat finna en approximation av ett nollställe.

Alternativet med `opts` använder vi då vi t.ex. vill ange hur noggrant lösningen skall beräknas. Man skapar vektorn `opts` med funktionen `optimset`, se hjälptexten.

Vi använder `fzero` för att beräkna en lösning till

$$f(x) = 0.5(x-2)^2 - 2\cos(2x) - 1.5 = 0$$

för en sista gång.

```
>> f=@(x)0.5*(x-2).^2-2*cos(2*x)-1.5;
>> x0=4;
>> x=fzero(f,x0)
x =
   3.866407887464427
```

Uppgift 4. Betrakta ekvationen

$$f(x) = \frac{3 + \sin(2x)}{1 + \mathrm{e}^{0.03x^2}} - 1.2 = 0$$

Rita graf och beräkna samtliga nollställen noggrant med `fzero`. Tänk på att använda elementvisa operationer.