Newtons metod

1 Inledning

Vi skall fortsätta med att lösa ekvationer. I förra veckan såg vi på intervallhalveringsmetoden. Den är pålitlig men ganska långsam. I denna vecka skall vi använda Newtons metod som är mycket snabbare, bara vi har en bra första approximation av en lösning.

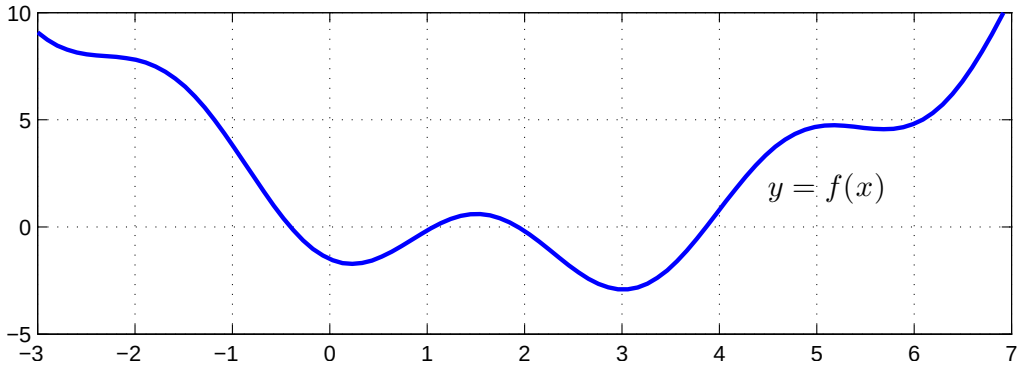
Som exempel kan vi ta,

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

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.

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.

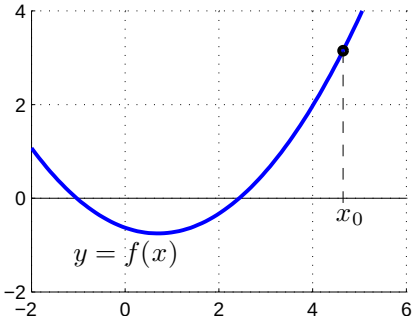
$$y = f(x_k) + f'(x_k)(x - x_k)$$

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

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

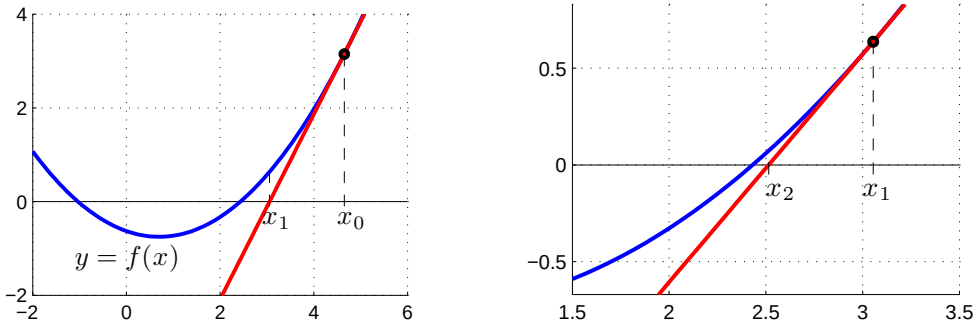
som en ny approximation av nollstället.

Vi ser på några steg med metoden: Starta med en approximation x_0 av ett nollställe 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 veckan 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. f=@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 ett programskal `min_newton.m` på MATLAB-hemsidan att utgå ifrån.

Uppgift 3. Pröva nu din funktion `min_newton` på följande ekvationer. Rita grafer och beräkna samtliga nollställen.

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

4 Färdigt program i MATLAB

Det finns en färdig funktion `fzero` för att lösa icke-linjära ekvationer. För en sista gång ser vi på exemplet från inledningen. Vi har alltså

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

och i MATLAB löser vi med

```
>> f=@(x)0.5*(x-2).^2-2*cos(2*x)-1.5;
>> x0=4;
>> x=fzero(f,x0)
x =
    3.8664
```

Leta gärna upp hjäptexten för funktionen `fzero` och läs lite.

Utskriften av beräkningsresultatet ovan gjordes med fem siffror. Vill vi få fler siffror utskrivna kan vi ge kommandot `format long` innan utskriften. Med `format short` får vi tillbaka den korta varianten. Så kallad scientific notation får vi med `format short e` respektive `format long e`.

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 komponentvisa operationer.

1 Målsättning

Avsikten med denna laboration är i stort sett samma som för förra laborationen som handlade om intervallhalveringsmetoden. Nu gäller det att förstå Newtons metod. Programmeringsarbetet är dock nästan samma. Avslutningsvis skall vi bekanta oss med `fzero`, ett färdigt program för ekvationslösning i MATLAB.

2 Kommentarer och förklaringar

Newtons metod utnyttjar funktions- och derivatavärden för att lokalt med en rät linje approximera den funktion vi söker nollställe till. Varje ny approximation ger en bättre bestämning av nollstället. Det finns dock en svag punkt, en rät linje är inte alltid en bra approximation i ett stort område. Vi måste därför alltid ge Newtons metod en bra första gissning av nollstället. Men det är inget större problem för vi skall ju ändå rita en graf så att vi får grepp om hur många nollställen det finns, var ungefär de ligger och vilka som är av intresse för oss.

Det färdiga programmet `fzero` bygger på Newtons metod (och till en del på intervallhalveringsmetoden). Derivatan av funktionen (som vi söker nollställe till) approximeras i programmet så vi behöver inte ge den.

3 Lärandemål

Efter denna laboration skall du kunna

- redogöra för den grundläggande idén i Newtons metod
- lösa ekvationer $f(x) = 0$, genom att beskriva f som en function i MATLAB, rita dess graf, grovt lokalisera nollställen av intresse och sedan bestämma dem noggrant med er funktion `min_newton`
- använda `fzero` för ekvationslösning