

Nollställena och extrempunkter

1 Inledning

Vi skall beräkna nollställena och extrempunkter till funktioner. För att se hur många nollställena en funktion har och ungefär var de ligger behöver vi rita en graf av funktionen. Samma gäller för extrempunkter, vi studerar funktionens graf för att se ungefär var extrempunkterna ligger och av vilken typ de är (max- eller minpunkt). Därefter beräknar vi noggrant de nollställena eller extrempunkter som vi är intresserade av med hjälp av i MATLAB inbyggda program. För att använda dessa program behöver vi även se på hur man definierar egna funktioner.

2 Grafitning

Vi ritade några grafer redan i första laborationen. Som exempel tar vi grafen till $f(x) = \sin(x)$ över intervallet $0 \leq x \leq 2\pi$. Denna kan vi rita upp i MATLAB enligt

```
>> x=linspace(0,2*pi);  
>> y=sin(x);  
>> plot(x,y)
```

Med funktionen `linspace` bildar vi en vektor med x -värden jämnt fördelade mellan 0 och 2π och med `sin` bildar vi en (lika stor) vektor med motsvarande sinusvärden. Dvs. alla sinusvärden för de olika x -värdena räknas ut samtidigt.

Som ytterligare ett exempel tar vi grafen till $f(x) = x \sin(x)$ över intervallet $0 \leq x \leq 15$. Vi ritar grafen med

```
>> x=linspace(0,15);  
>> y=x.*sin(x);  
>> plot(x,y)
```

Multiplikationen `.*` kallas elementvis multiplikation. Radvektorerne `x` och `sin(x)` skall ju multipliceras ihop elementvis så att $y_1 = x_1 \sin(x_1)$, $y_2 = x_2 \sin(x_2)$, $y_3 = x_3 \sin(x_3)$, osv.

Utöver elementvis multiplikation `.*` finns elementvis division `./` och upphöjt till `.^` som fungerar på motsvarande sätt.

Ibland vill man rita flera grafer i samma koordinatsystem. Efter att ha ritat första grafen ger man kommandot `hold on` för att bevara den, sedan kan man rita fler grafer ovanpå tills man tar bort skyddet med `hold off`. Med `axis` kan vi sätta lämpliga skalor i koordinatsystemet.

Vi påminner oss om att vi kan lägga på ett rutnät med `grid on` och ta bort det igen med `grid off`, om vi vill det. Med `xlabel` och `ylabel` kan vi sätta texter på axlarna och med `title` kan vi sätta rubrik på koordinatsystemet.

Uppgift 1. Rita grafen till $f(x) = x - x \cos(7x)$ över intervallet $0 \leq x \leq 8$.

3 Egna funktioner

När vi skall beräkna nollställena och extrempunkter behöver vi både rita grafer av den funktion vi studerar och ge en beskrivning av den till de program vi skall använda. Som exempel kan vi ta funktionen $f(x) = x \sin(x)$. Vi kan i MATLAB beskriva denna funktion med

```
>> f=@(x)x.*sin(x);
```

Vi påminner oss om att detta kallas för en anonym funktion och $@(x)$ är det som anger att det är just en funktion. Satsen ovan är en definition, vi talar om hur funktionen skall räknas ut men ingen beräkning görs. Dvs. en ren beskrivning av hur funktionen är uppbyggd.

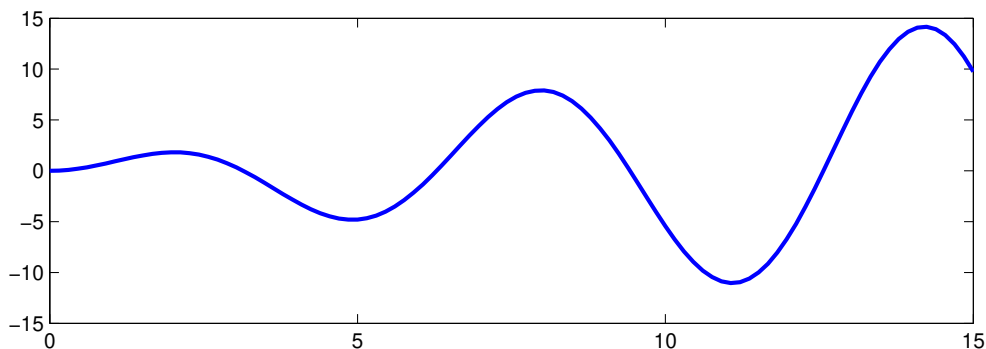
Vill vi nu rita grafen till $f(x)$ över intervallet $0 \leq x \leq 15$ gör vi det enligt

```
>> x=linspace(0,15);  
>> y=f(x);  
>> plot(x,y)
```

alternativt lite enklare

```
>> x=linspace(0,15);  
>> plot(x,f(x))
```

och här ser vi resultatet.



Uppgift 2. Beskriv följande funktion i MATLAB

$$f(x) = \cos(x) - \exp(-x^2/2) \sin(5x)$$

och rita dess graf över intervallet $-2\pi \leq x \leq 2\pi$.

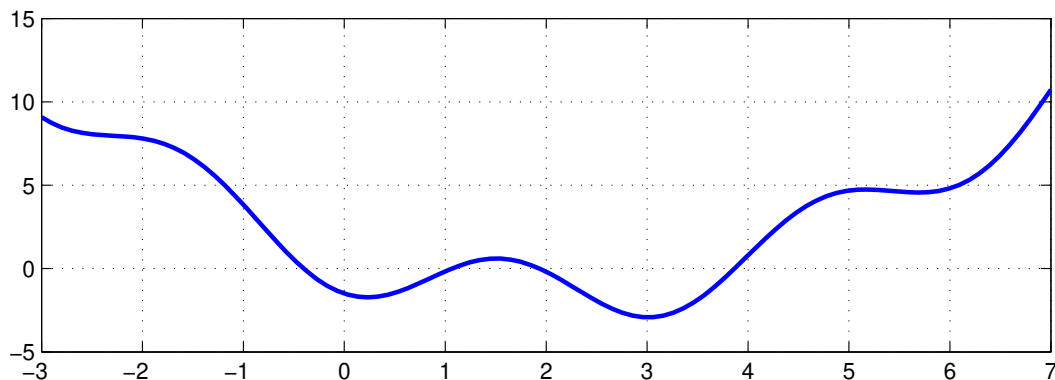
4 Nollställena

Vi skall lösa ekvationer, dvs. beräkna nollställena till funktioner. Som exempel tar vi ekvationen

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

Först beskriver vi funktionen och ritar dess graf så att vi ser hur många nollställena det finns 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))
>> grid on
```



Vi ser lösningar till $f(x) = 0$ som de punkter där grafen skär x -axeln.

Om vi inte kan få fram någon formel för att lösa en ekvation så kan vi beräkna en approximation med t.ex. **Newtons metod**: Antag att x_k är en approximation av ett nollställe till funktionen $f(x)$. Följ tangenten i punkten $(x_k, f(x_k))$, dvs.

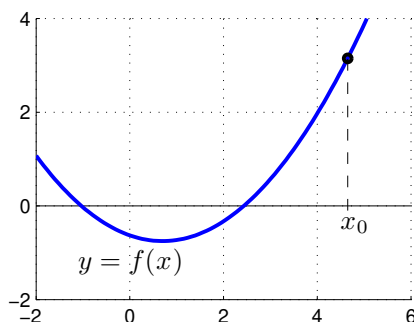
$$y = f(x_k) + f'(x_k)(x - x_k)$$

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

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

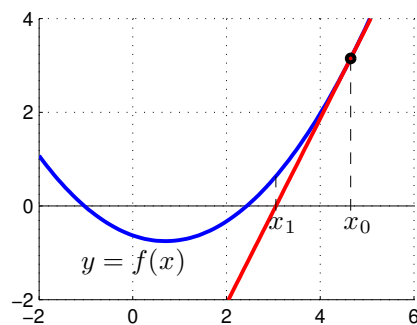
som en ny approximation av nollstället.

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



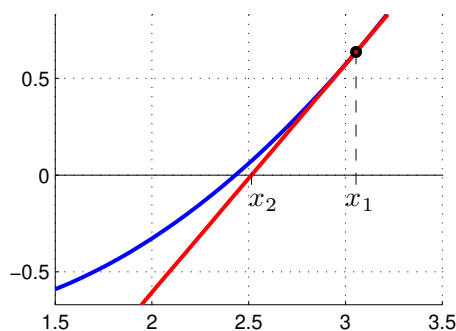
Bilda tangenten $y = f(x_0) + f'(x_0)(x - x_0)$ till f i $x = x_0$ och tag dess skärningspunkt med x -axeln som en ny approximation

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



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 än åtta korrekta decimaler (felet mindre än $\frac{1}{2} \times 10^{-8}$).

Uppgift 3a. Kopiera koden i exemplet ovan och testkör den. Hur många varv utför **for**-loopen? Ta sedan bort (eller kommentera bort) raderna

```

if abs(h)<tol
    break
end

```

och testkör koden igen. Hur många varv utför **for**-loopen? Slå upp **if**-sats och **break** i hjälptexterna (skriv **help if** respektive **help break** i kommandofönstret) Försök förklara (med ord) vad som avbryter **for**-loopen.

Uppgift 3b. Låt $f(x) = x^3 - \cos(4x)$. Lös ekvationen $f(x) = 0$. Rita först 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. Använd dig av koden i exemplet ovan när du löser uppgiften (dvs. kopiera koden och modifiera/utöka den så att den löser uppgiften).

Som bekant finns det ett färdigt kommando, **fzero** i MATLAB som beräknar nollställena till en given funktion. Slå upp kommandot i MATLABs dokumentation (skriv **doc fzero** i kommandofönstret), eller titta i tidigare laboration, så att du ser hur kommandot anropas.

Uppgift 4. 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 **fzero**. Upprepa tills du beräknat alla lösningar till ekvationen.

5 Extrempunkter

Vi skall bestämma max- och minpunkter till en funktion $f(x)$. Som exempel tar vi

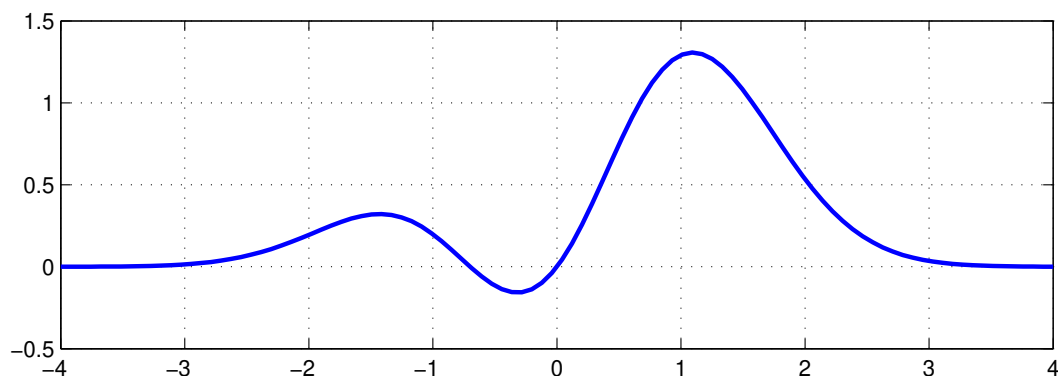
$$f(x) = (x + 1.5x^2 + 0.1x^3) \exp(-0.7x^2)$$

Vi ritar upp grafen för att se var de stationära punkterna finns och av vilken typ de är.

```

>> f=@(x) (x+1.5*x.^2+0.1*x.^3).*exp(-0.7*x.^2)
>> x=linspace(-4,4);
>> plot(x,f(x))
>> grid on

```



Det ser ut som vi har tre stationära punkter, en lokal minpunkt (global minpunkt) och två lokala maxpunkter (varav en är en global maxpunkt).

Vill vi beräkna extrempunkter till en funktion $f(x)$ som är deriverbar kan vi beräkna stationära punkter genom att lösa ekvationen $g(x) = f'(x) = 0$. Sedan får vi kontrollera om dessa är extrempunkter och i så fall av vilken typ.

Det finns också metoder som söker lokalt minimum till en funktionen utan att använda derivata. I MATLAB finns funktionen `fminbnd` som finner lokal minpunkt till en given funktion och används enligt `x=fminbnd(fun,x1,x2)`. Här beskriver `fun` funktionen vi skall minimera, `x1` och `x2` ger ett intervall som omsluter minpunkten vi söker.

Vill vi istället maximera funktionen så finner vi maxpunkt genom att minimera $h(x) = -f(x)$.

Från grafen till funktionen ser vi att det finns en lokal minpunkt i intervallet $-0.5 \leq x \leq 0$ och lokala maxpunkter i intervallen $-1.6 \leq x \leq -1.2$ och $1 \leq x \leq 1.5$.

Vi bestämmer den lokala minpunkten med

```
>> x=fminbnd(f,-0.5,0)
x =
    -0.3183
```

Vi bestämmer nu de lokala maxpunkterna, genom att minimera $h(x) = -f(x)$, med

```
>> h=@(x)-f(x);
>> x=fminbnd(h,-1.6,-1.2)
x =
    -1.4268

>> x=fminbnd(h,1,1.5)
x =
    1.0960
```

Uppgift 5. Bestäm samtliga extrempunkter till funktionen

$$f(x) = \frac{1 + x^2 - 1.5x^3 + 0.5x^4}{1 + x^4}$$

Rita graf och använd `fminbnd` på lämpligt sätt. Ange av vilken typ extrempunkterna är och beräkna eventuella max- och minvärden. Kom ihåg att använda elementvisa operationer.