Matematikk 1000

Øvingsoppgaver i numerikk – leksjon 7

Løsningsforslag

Oppgave 1 – Newtons metode

a) Newtons metode kan implementeres slik:

```
1  % Skript som implementerer Newtons metode for
2  % likninga f(x)=0.
3
4  x0=1;             % Velger en x_0
5  Pres=1e-4;        % Bestemmer ønsket nøyaktighet
6
7  xGammel=1e2;      % Initierer 'gammel x' og setter en høy verdi
8  x=x0;             % Setter x lik x_0
9
10 %While-løkke som kjører så lenge differansen er større enn Pres
11 while abs(x-xGammel)>Pres
12   xGammel=x;      % Tilordner den forrige x-verdien til xGml
13   x=x-(x-cos(x))/(1+sin(x));   % Regner ut ny x
14 end
15 Loesning=x        % Skriver løsning til skjerm
```

Her har vi valgt å starte med $x_0 = 1$, og vi krever en nøyaktighet på 10^{-4}. Vi kjører skriptet, som jeg har kalt "`NewtonsMetode.m`":

```
> NewtonsMetode
Loesning =  0.73909
```

Vi har funnet løsninga $\underline{x = 0.73909}$.

b) Selvsagt kan vi, som i leksjon 5, bestemme antall iterasjoner ved å telle hvor mange ganger `x` blir skrevet til skjerm når vi fjerner semikolonet i linje 13. Men som vi har sett før, kan vi kan gjøre det litt mer elegant. Her har tre linjer, linje 9, 15 og 18, blitt lagt til skriptet:

```
1  % Skript som implementerer Newtons metode for
2  % likninga f(x)=0.
3
4  x0=1;            % Velger en x_0
5  Pres=1e-4;       % Bestemmer ønsket nøyaktighet
6
7  xGammel=1e2;     % Initierer 'gammel x' og setter en høy verdi
8  x=x0;            % Setter x lik x_0
9  its=0;           % its er antall iterasjoner, satt til 0 først
10
11 %While-løkke som kjører så lenge differansen er større enn Pres
12 while abs(x-xGammel)>Pres
13   xGammel=x;     % Tilordner den forrige x-verdien til xGml
14   x=x-(x-cos(x))/(1+sin(x));   % Regner ut ny x
15   its=its+1;     % Antall iterasjoner økes med én
16 end
17 Loesning=x       % Skriver løsning til skjerm
18 Iterasjoner=its  % Skriver antall iterasjoner til skjerm
```

Om vi nå kjører skriptet om igjen, får vi:

```
> NewtonsMetode
Loesning =  0.73909
Iterasjoner =  3
```

Med "`Pres=1e-1`" (linje 5), får vi

```
> NewtonsMetode
Loesning =  0.73911
Iterasjoner =  2
```

og om vi endrer `Pres` til 10^{-8}, får vi:

```
> NewtonsMetode
Loesning =  0.73909
Iterasjoner =  4
```

Generelt behøver vi flere iterasjoner jo større nøyaktighet vi krever. Men en nøyaktighet på 10^{-8} etter 4 iterasjoner er slett ikke ille[1]!

Til sammenligning behøvde vi 14 iterasjoner for å komme fram til et svar med en nøyaktighet som krevde 3 iterasjoner med Newtons metode.

c) Om vi velger $x_0 = 3\pi/2$, vil metoden bryte sammen fordi $f'(x_0) = 0$. I MATLAB ser feilmeldinga slik ut:

[1] Vi minner om at MATLAB opererer med flere desimaler enn de 5 som har blitt skrevet til skjerm her; hvis ikke ville det gitt lite mening å snakke om en nøyaktighet på 10^{-8}.

```
>> NewtonsMetode

Loesning =
   NaN

Iterasjoner =
     2
```

Man kan jo lure på hvorfor `while`-løkka har kjørt to ganger her. Noen løsning av likninga får vi i alle fall ikke ('`NaN` står for *not a number*; slikt dukker for eksempel opp når vi prøver å dele på 0...'). Om vi velger x_0 nær men ikke lik $3\pi/2$, for eksempel med $x_0 = 4.7$, og setter `Pres` til å være 10^{-4}, får vi

```
>> NewtonsMetode

Loesning =
    0.7391

Iterasjoner =
   214
```

Her behøver vi altså hele 214 iterasjoner! Hva sjedde? Figur 1 er ment å illustrere dette. Av figuren ser vi at x_1 havner langt fra nullpunktet. Derfor vil vi også behøve mange iterasjoner for å nærme oss nullpunktet igjen. [Denne figuren må erstattast av ein som viser skjæringspunktet også for den neste tangenten.]

Oppgave 2 – Mer Newtons metode

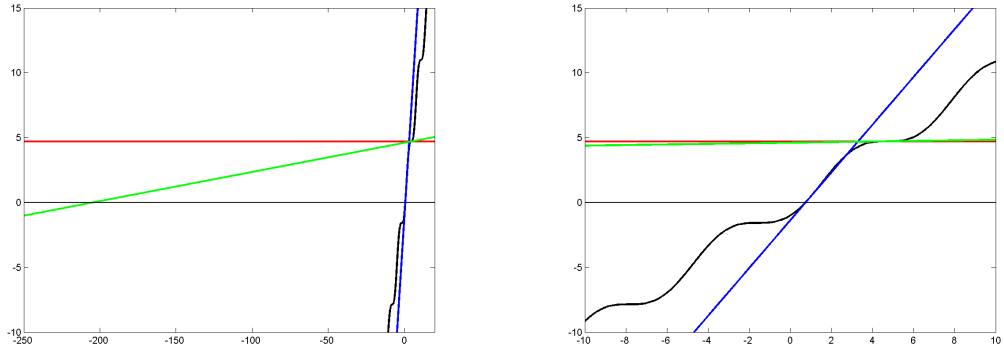
3.4: 2

a)

$$\begin{aligned} f(x) &= x^3 + 3x + 9 \\ f'(x) &= 3x^2 + 3 > 0 \quad \text{for alle} \quad x \quad . \end{aligned}$$

$f(-2) = -5 < 0$, $f(0) = 9 > 0$. Siden f er kontinuerlig, må funksjonen da ha minst ett nullpunkt på intervallet $[-2, 0]$ ved skjæringssetninga. Siden f' alltid er positiv, kan f heller ikke flere enn ett nullpunkt.

Vi gjør noen små endringer på skriptet over:



Figur 1: Plottet viser $f(x)$ (svart) sammen med tangenten for $x_0 = 1$ (blå), for $x = 2\pi/2$ (grønn) og for $x_0 = 4.5$ (rød). Vi ser at den blå tangenten skjærer x-aksen omtrent på det samme stedet som $f(x)$ gjør det. Den grønne tangenten, derimot, er horisontal. Også den blå tangenten er nesten horisontal ($f'(4.0) \approx 0$) slik at x_1 havner svært langt frå x_0 og også løsninga av likninga. Dermed vil Newtons metode konvergere sakte med et slikt valg av x_0 – om den vil konvergere i det hele. De to plottene viser det samme funksjonane på litt ulike skalaer.

```
1  % Skript som implementerer Newtons metode for
2  % likninga f(x)=0.
3
4  x0=-1;             %Velger en x_0
5  Pres=1e-4;         %Bestemmer ønsket nøyaktighet
6
7  xGammel=1e2;       %Initierer 'gammel x' og setter en høy verdi
8  x=x0;              %Setter x lik x_0
9  its=0;             % its er antall iterasjoner, satt til 0 først
10
11 %While-løkke som kjører så lenge differansen er større enn Pres
12 while abs(x-xGammel)>Pres
13   xGammel=x;       %Tilordner den forrige x-verdien til xGml
14   x=x-(x^3+3*x+9)/(3*x^2+3);   % Regner ut ny x
15   its=its+1;       % Antall iterasjoner økes med én
16 end
17 Loesning=x         %Skriver løsning til skjerm
18 Iterasjoner=its    % Skriver antall iterasjoner til skjerm
```

Vi har her justert x_0 (linje 4) og funksjonsuttrykket (linje 14). I MATLAB-vinduet får vi:

```
> NewtonsMetode
Loesning = -1.6097
Iterasjoner =  5
```

Altså: $\underline{x \approx -1.61}$.

b)

$$\begin{aligned} f(x) &= \tan x - x + 1, \quad D_f = (-\pi/2, \pi/2) \\ f'(x) &= \frac{1}{\cos^2 x} - 1 \geq 0 \quad \text{for alle} \quad x \end{aligned}$$

$f(0) = 1 > 0$, $f(-1.5) \approx -11.6 < 0$. Som over, vil f ha ett, og bare ett, nullpunkt ved skjæringssetninga og på grunn av at f' er strengt økende. Vi lar x_0 være -1 og justerer linje 14 til

```
x=x-(tan(x)-x+1)/(1/(cos(x))^2-1);
```

MATLAB gir

```
> NewtonsMetode
Loesning = -1.1323
Iterasjoner =  5
```

Altså: $\underline{x \approx -1.13}$.

c)

$$\begin{aligned} f(x) &= x + \sin x + 1 \\ f'(x) &= 1 + \cos x \geq 0 \quad \text{for alle} \quad x \end{aligned}$$

$f(-\pi/2) = -\pi/2 + 0 + 1 = -(\pi/2 - 1) < 0$, $f(0) = 1 > 0$. Av samme grunner som over, har f ett, og bare ett, nullpunkt. Dette nullpunktet vil ligge i intervallet $[-\pi/2, 0]$. Vi fortsetter å la x_0 være -1 og justerer linje 14 i skriptet til

```
x=x-(x+sin(x)+1)/(1+cos(x));
```

Vi kjører skriptet vårt i MATLAB og får:

```
> NewtonsMetode
Loesning = -0.51097
Iterasjoner =  4
> x=Loesning ;
> x+sin(x)+1
ans = 1.1102e-016
```

$\underline{x \approx -0.51}$. Her har vi også kontrollert at $x + \sin x + 1$ faktisk er rimelig nære 0 for den x-verdien vi kom fram til.

3.4: 6

a) Skjæringspunktet finner vi ved løse likninga

$$\begin{aligned} f(x) &= g(x) \\ x^3 &= e^{-x} \\ x^3 - e^{-x} &= 0 \end{aligned}$$

Den deriverte av høgre side er $3x^2 + e^{-x}$. Vi velger $x_0 = 0$, endrer den nå velkjente linja i skriptet til

```
x=x-(x^3-exp(-x))/(3*x^2+exp(-x));
```

og setter parameteren `Pres` til $5 \cdot 10^{-4}$, siden vi skal finne svaret med tre riktige desimaler. I MATLAB får vi svaret $\underline{x = 0.773}$ etter 5 iterasjoner.

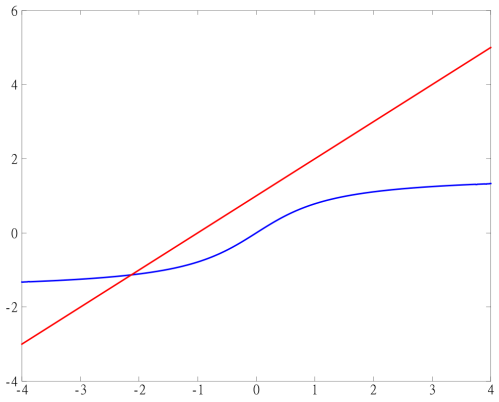
b)

$$\begin{aligned} f(x) &= g(x) \\ \arctan x &= x + 1 \\ \arctan x - x - 1 &= 0 \end{aligned}$$

Den deriverte av høgre side er

$$\frac{1}{1+x^2} - 1 \quad .$$

For å få en viss ide om hvilken x_0 det kan være lurt å velge, plotter vi f og g – se figur 2. Skjæringspunktet ser ut til å ha x-verdi nær -2, så vi



Figur 2: Plot av funksjonene $f(x) = \arctan x$ (blå) og $g(x) = x + 1$ (rød). De ser ut til å skjære hverandre for $x \approx -2$.

velger dette som x_0. Vi justerer skriptet:

```
x=x-(atan(x)-x-1)/(1/(1+x^2)-1);
```

og beholder valget av nøyatighet fra deloppgave a). Vi får svaret $\underline{x = -2.132}$ etter 3 iterasjoner.

Her kan det være naturlig å preisisere at det kanskje hadde vært smidigere å la skriptet referere til ei funksjonsfil tilsvarende den opprinnelige rekursjonsformelen.

Ekstra: Oppgave 3 – Attraktive fikspunkter

Vi kan nok med fordel ta utgangspunkt i skriptet vi lagde for Netwons metode. For å løse likninga $x = \cos x$, kan skjemaet implementeres slik:

```
1  % Skript som estimerer løsning av
2  % likninga x=f(x) ved iterasjon
3
4  x0=1;             % Velger en x_0
5  Pres=1e-4;        % Bestemmer ønsket nøyaktighet
6
7  xGammel=1e2;      % Initierer 'gammel x' og setter en høy verdi
8  x=x0;             % Setter x lik x_0
9  its=0;            % its er antall iterasjoner, satt til 0 først
10
11 % While-løkke som kjører så lenge differansen er større enn Pres
12 while abs(x-xGammel)>Pres
13   xGammel=x;      % Tilordner den forrige x-verdien til xGml
14   x=cos(x);       % Regner ut ny x
15   its=its+1;      % Antall iterasjoner økes med én
16 end
17 Loesning=x        % Skriver løsning til skjerm
18 Iterasjoner=its   % Skriver antall iterasjoner til skjerm
```

Som du ser, er skriptet svært likt det vi kjenner frå før. Men der er en svakhet, som vi forsåvidt implementeringa av Newtons metode også led under: Dersom x ikke nærmer seg noen spesiell verdi, dersom x ikke *konvergerer*, vil `while`-løkka aldri stoppe. Grunnen til at dette er et større problem her enn i de forrige oppgavene, er at det hender langt oftere med denne metoden at det ikke konvergerer mot noen spesiell løsning enn med Newtons metode. Som nevnt tideligere, dersom det skulle skje at vi setter i gang ei "uendelig" `while`-løkke, kan denne brytes ved å holde `Ctrl`-tasten nede og trykke `C`.

Vi velger å justere `while`-løkka slik at den uansett stopper etter 100 iterasjoner. Fra og med linje 11 erstatter vi skriptet med følgende:

```
% While-løkke som kjører så lenge differansen er større enn Pres
% eller antall iterasjoner er mindre enn 100
while abs(x-xGammel)>Pres & its<100
  xGammel=x;     % Tilordner den forrige x-verdien til xGml
  x=cos(x);      % Regner ut ny x
  its=its+1;     % Antall iterasjoner økes med én
end

if its==100       % Skriver negativt resultat til skjerm
  'Ikke konvergert etter 100 iterasjoner'
else
  Loesning=x      % Skriver løsning til skjerm
  Iterasjoner=its % Skriver antall iterasjoner til skjerm
end
```

`while`-løkka kjører så lenge $|x_{n+1} - x_n|$ er større enn den presisjonen vi har satt *og* antall iterasjoner er mindre enn 100; den stopper når $|x_{n+1} - x_n|$ blir mindre enn `Pres` *eller* antall iterasjoner når 100.

Vi har kalt skriptet "`FiksPunkt.m`". Det viser seg at metoden konvergerer – men slett ikke like raskt som Newtons metode:

```
> FiksPunkt
Loesning =  0.73905
Iterasjoner =  23
```

La oss også se på oppgave 6, delkapittel 3.4 i læreboka. I deloppgave a) skulle vi løse likninga $x^3 = e^{-x}$. Denne kan skrives på forma $x = f(x)$ på to måter:

$$x = \sqrt[3]{e^{-x}} = e^{-x/3} \quad \text{eller} \quad x = -\ln x^3 = -3\ln x \quad .$$

Vi tester begge. I det første tilfellet endrer vi funksjonsuttrykket i den relevante linja til til

```
  x=exp(-x/3);    % Regner ut ny x
```

Vi beholder x_0 og `Pres` og kjører skriptet:

```
> FiksPunkt
Loesning =  0.77287
Iterasjoner =  7
```

I tilfelle to, ser tilsvarende linje slik ut:

```
  x=-3*log(x);    % Regner ut ny x
```

I kommandovinduet får vi:

```
> FiksPunkt
Loesning = -Inf - 9.425i
Iterasjoner =  4
```

Dette har tydeligvis ikke gått bra i det hele tatt. Det går ikke noe bedre om vi endrer x_0 til 0.5:

```
> FiksPunkt
ans = Ikke konvergert etter 100 iterasjoner
```

Vi prøver oss på deloppgave b) også:

$$\arctan x = x + 1 \Leftrightarrow x = \tan(x+1) \Leftrightarrow x = \arctan x - 1$$

Vi impementerer den første varianten først:

```
  x=tan(x+1);      % Regner ut ny x
```

Vi setter $x_0 = -2$ og kjører skriptet:

```
> FiksPunkt
ans = Ikke konvergert etter 100 iterasjoner
```

Den andre varianten, `x=atan(xGml)-1;`, fungerer langt bedre:

```
> FiksPunkt
Loesning = -2.1323
Iterasjoner =  6
```

Erfaringene vi gjør oss her, stemmer nok generelt: Newtons metode er vanligvis mer pålitelig enn denne metoden. Det skal også legges til at det finnes kriterier for når denne metoden virker og ikke. Vi skal ikke gå inn på disse her.