Mathias Bruun Houmøller - 202006837

Opgave 2 i "M3NUM1 F15 Ordinær eksamen"

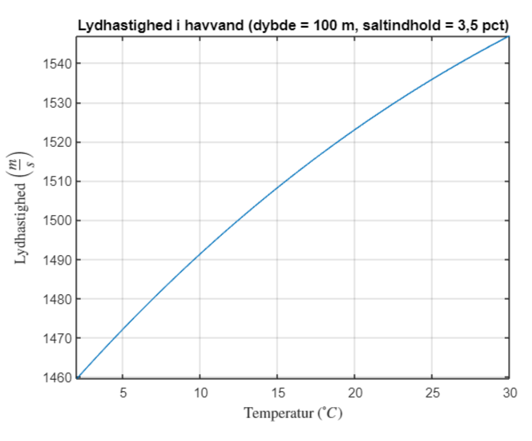
a)

Lydhastigheden, c [m/s], i havvand varierer med temperaturen, T [°C]. I havvand med 3,5% salt vil der på 100 meters dybde gælde følgende sammenhæng:

$$c(T) = 0{,}0002374T^3 - 0{,}05304T^2 + 4{,}591T + 1450{,}59$$

> Benyt MATLAB til at plotte funktionen $c(T)$ i intervallet $2°C \le T \le 30°C$. Plottet skal forsynes med *grid* samt titlen Lydhastighed i havvand (dybde = 100 m, saltindhold = 3,5 pct). Betegnelserne på den vandrette og lodrette akse skal være henholdsvis Temperatur (grader C) og Lydhastighed (m/s). Besvarelsen skal indeholde såvel de benyttede MATLAB-kommandoer som det resulterende plot.

```
%Funktionen defineres
f = @(T) 0.0002374.*T.^3 - 0.05304.*T.^2 + 4.591.*T + 1450.59;

%temperatur intervallet der ses på
Temp = [2,30]; %degreeC

fplot(f,Temp), grid; %Plotter funktionen i temperatur intervallet , yline(1500, "--")
%sætter titlen på plottet
title("Lydhastighed i havvand (dybde = 100 m, saltindhold = 3,5 pct)")

%sætte teksten på x-aksen
xlabel("Temperatur ($^\circ C$)",'Interpreter','latex')

%sætte teksten på y-aksen
ylabel("Lydhastighed $\left(\frac{m}{s}\right)$",'Interpreter','latex')
```

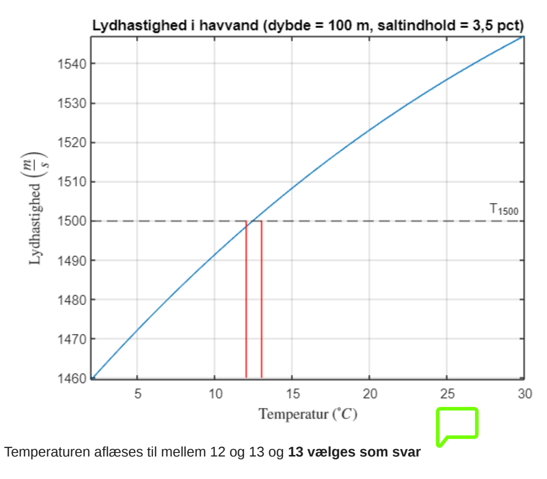


b)

Den temperatur, hvor lydhastigheden er 1500 m/s, benævnes i det følgende T_{1500}.

> Bestem T_{1500} visuelt ved aflæsning på plottet udarbejdet i spørgsmål (a). Der ønskes et heltalligt svar.

```
yline(1500,'--','T_{1500}'); % laver en linje ved hastigheden 1500 for bedre aflæsning
line([12 12], [1460 1500], color="r"); %aflæst resultat
line([13 13], [1460 1500], color="r"); %aflæst resultat
```



Temperaturen aflæses til mellem 12 og 13 og **13 vælges som svar**

c)

Her følger et delvist færdigt MATLAB-script, der anvender bisektionsmetoden til at bestemme T_{1500} med større nøjagtighed:

```
f = @(T) 0.0002374*T^3 - 0.05304*T^2 + 4.591*T + 1450.59 - <UDFYLD>;
Tl = 10;      % Nedre intervalendepunkt (grader C)
Tu = 15;      % Øvre intervalendepunkt (grader C)
es = 0.01;    % Ønsket nøjagtighed af rod (%)
maksit = 20;  % Maksimal antal iterationer
it = 0;       % Startværdi for iterationstælleren it
while 1
  Tr = (<UDFYLD>)/2;
  ea = abs(Tr-Tl)/Tr*100;
  it = it + <UDFYLD>;
  if f(Tl)*f(Tr) > 0
    <UDFYLD> = Tr;
  elseif f(Tl)*f(Tr) < 0
    <UDFYLD> = Tr;
  else
    ea = 0;
  end
  if <UDFYLD> <= es || it >= <UDFYLD>, break, end
end
T1500 = Tr
ea
it
```

Scriptet kan downloades fra Blackboard under navnet `bisektion.m`.

> Færdiggør MATLAB-scriptet ved at erstatte tekstangivelserne **<UDFYLD>** med korrekt MATLAB-kode, og kør derefter scriptet. I besvarelsen ønskes såvel det færdiggjorte script som dets output i kommandovinduet anført, og det ønskes forklaret, hvad slutværdierne af `ea` og `it` betyder.

```
% bisektion
f = @(T) 0.0002374*T^3 - 0.05304*T^2 + 4.591*T + 1450.59 - 1500;
Tl = 10; % Nedre intervalendepunkt (grader C)
Tu = 15; % Øvre intervalendepunkt (grader C)
es = 0.01; % Ønsket nøjagtighed af rod (%)
maksit = 20; % Maksimal antal iterationer
it = 0; % Startværdi for iterationstælleren it
while 1
    Tr = (Tl + Tu)/2; %T værdien i roden
    ea = abs(Tr - Tl)/Tr*100; %usikkerheden
    it = it + 1; %antal iterationer
    if f(Tl)*f(Tr) > 0
        Tu = Tr; %Der sættes nyt upper limit
    elseif f(Tl)*f(Tr) < 0
        Tl = Tr; %Der sættes nyt lower limit
    else
        ea = 0; %usikkerheden er 0 da vi har ramt rigtigt
    end
    %Forloopet brydes hvis vi når et maksimalt antal forsøg eller vi kommer
    %indenfor usikkerheden.
    if ea <= es || it >= maksit, break, end
end
T1500 = Tr % Temperaturen printes
```

```
T1500 = 12.5012
```

```
ea % Usikkerheden printes
```

```
ea = 0.0098
```

```
it % Antal interationer der er gennemgået
```

```
it = 12
```

ea er usikkerheden

it er antal iterationer(forsøg)

d)

Bestem T_{1500} ved hjælp af Newton-Raphsons metode, idet der benyttes en startværdi på 15. Der skal udføres så mange iterationer, at den approksimative fejl på den fundne værdi af T_{1500} er højest 0,01%. I besvarelsen ønskes såvel resultater som beregningsformler for de enkelte iterationer anført (gerne på tabelform).

newtraph.m funktionen følger:

```
function [root,ea,iter]=newtraph(func,xr,es,maxit)
% newtraph: Newton-Raphson root location zeroes
%   [root,ea,iter]=newtraph(func,xr,es,maxit):
%   uses Newton-Raphson method to find the root of func
% input:
%   func = name of function symbolic
%   xr = initial guess
%   es = desired relative error (default = 0.0001%)
%   maxit = maximum allowable iterations (default = 50)
% output:
%   root = real root
%   ea = approximate relative error (%)
%   iter = number of iterations

if nargin<2 %Hvis der er mindre end 2 input gives der en advarsel
    error('at least 2 input arguments required')
end

if nargin<3||isempty(es) %sætter default value hvis ikke intastes
    es=0.0001;
end

if nargin<4||isempty(maxit) %sætter default value hvis ikke intastes
    maxit=50;
end

iter = 0; %antal iteration starter på 0
%laver tomme arrays til tabel udskrift
Resultat = [];
f_list = [];
df_list = [];
Error = [];

%differentier funktionen symbolsk og laver den numerisk igen
f = matlabFunction(func);
df = matlabFunction(diff(func));

while 1  %starter while loop
  xrold = xr; %gemmer den gamle xr værdi
  xr = xr - f(xr)/df(xr); %udregner roden.

%   syms T %sætter T som et symbol til visning af ligningen
%   f_show = xr - f(T)/df(T) %ligningen der bruges
  iter = iter + 1; % antal gange der er kørt igennem

  Resultat(end+1) = xr; %tilføjer roden til resultat listen
  f_list(end+1) = f(xrold); %tilføjer ligningen til lignings listen
  df_list(end+1) = df(xrold);

  if xr ~= 0 %Hvis xr ikke er 0 sættes en ny error således undgår vi at
      % dividere med 0
      ea = abs((xr - xrold)/xr) * 100;
  end
  Error(end+1) = ea;  %tilføjer error til resultat error listen

  %while loopet stoppes hvis vi når inden for vores satte stat error eller
  %vi når det maximale antal iterrationer
  if ea <= es || iter >= maxit
      break
  end
end
%Slut værdierne udprintes
root = xr
ea
iter

%Der laves en tabel med alle værdierne der er opnået
tabel = table(Resultat', f_list', df_list', Error',...
    VariableNames={'Rod(xr)','f(xr)', 'df/dT (xr)', ...
    'Error(ea)'})
end
```

```
syms T %laver T til et symbol
%sætter funktionen der skal undersøges denne skal sættes symbolsk
f = 0.0002374.*T.^3 - 0.05304.*T.^2 + 4.591.*T + 1450.59 - 1500;
vpa(diff(f),8) %Vi udskriver den differentierede funktion til excel
```

ans = $0.0007122\,T^2 - 0.10608\,T + 4.591$

```
newraph(f, 15, 0.0001, 50) %kalder ligningen
```

```
root = 12.4545
ea = 1.0758e-09
iter = 4
tabel = 4×4 table
```

	Rod(xr)	f(xr)	df/dT (xr)	Error(ea)
1	12.3664	8.3222	3.1600	21.2962
2	12.4544	-0.2981	3.3881	0.7065
3	12.4545	-0.0003	3.3803	0.0008
4	12.4545	-0	3.3803	0

```
ans = 12.4545
```

Med excel fås følgende

	A	B	C	D	E	F
1	Iteration	roden(xr)	f(x)	f'(x)	Error(ea)	
2	0	15,0000				Start input(xr)
3	1	12,3664	8,3222	3,1600	21,2962	
4	2	12,4544	-0,2981	3,3881	0,7065	
5	3	12,4545	-0,0003	3,3803	0,0008	
6	4	12,4545	0,0000	3,3803	0,0000	
7	5	12,4545	0,0000	3,3803	0,0000	

	A	B	C	D	E
1	Iteration	roden(xr)	f(x)	f'(x)	Error(ea)
2	0	15			
3	1	=B2-(C3/D3)	=0,0002374*B2^3-0,05304*B2^2+4,591*B2+1450,59-1500	=0,0007122*B2^2 - 0,10608*B2 + 4,591	=ABS(((B3-B2)/(B3)))*100
4	2	=B3-(C4/D4)	=0,0002374*B3^3-0,05304*B3^2+4,591*B3+1450,59-1500	=0,0007122*B3^2 - 0,10608*B3 + 4,591	=ABS(((B4-B3)/(B4)))*100
5	3	=B4-(C5/D5)	=0,0002374*B4^3-0,05304*B4^2+4,591*B4+1450,59-1500	=0,0007122*B4^2 - 0,10608*B4 + 4,591	=ABS(((B5-B4)/(B5)))*100
6	4	=B5-(C6/D6)	=0,0002374*B5^3-0,05304*B5^2+4,591*B5+1450,59-1500	=0,0007122*B5^2 - 0,10608*B5 + 4,591	=ABS(((B6-B5)/(B6)))*100
7	5	=B6-(C7/D7)	=0,0002374*B6^3-0,05304*B6^2+4,591*B6+1450,59-1500	=0,0007122*B6^2 - 0,10608*B6 + 4,591	=ABS(((B7-B6)/(B7)))*100

e)

Benyt den indbyggede MATLAB-funktion `fzero` til at bestemme T_{1500}. Besvarelsen skal indeholde såvel de benyttede MATLAB-kommandoer som resultatet, dvs. den fundne værdi af T_{1500}.

```
% help fzero
f = @(T) 0.0002374.*T.^3 - 0.05304.*T.^2 + 4.591.*T + 1450.59 - 1500; %definer vores funktion

fzero(f,12) %finder der hvor funktionen er 0 ved at indsætte funktionen og et start gæt som her er 12
```

```
ans = 12.4545
```

f)

Benyt den indbyggede MATLAB-funktion roots til at bestemme T1500. Besvarelsen skal indeholde såvel de benyttede MATLAB-kommandoer som resultatet, dvs. den fundne værdi af T1500.

```
% help roots
C = [0.0002374 -0.05304 4.591 1450.59-1500]; % array med constanterne i polynomiet
roots(C) % finder rødderne af polynomiet når konstanterne er gevet %polyval(C,12.4545) kan tage et polynomie og indsæt en x værdi.
```

```
ans = 3×1 complex

10^2 ×
   1.0548 + 0.7473i
   1.0548 - 0.7473i
   0.1245 + 0.0000i
```

Opgave 2 i "M3NUM1 E15 Reeksamen"

Denne opgave går ud på at finde rødder i følgende funktion:

$$f(x) = 300\cos(2x) + x^3 - 14x^2 - 36x + 150$$

a)

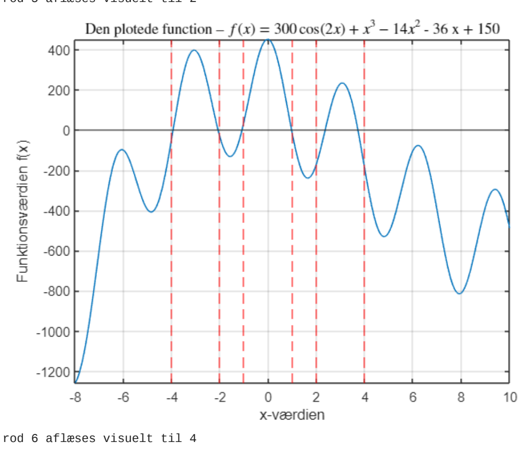
Benyt MATLAB til at plotte $f(x)$ i intervallet x = -8 til 10. Plottet skal forsynes med *grid* samt passende betegnelser på akserne. Bestem rødderne visuelt vha. plottet (som heltal). Besvarelsen skal indeholde plottet samt de benyttede MATLAB-kommandoer til frembringelse af plottet. Besvarelsen skal desuden indeholde de aflæste rødder.

```
clear % Tidliger variable rydes
f = @(x) 300 .* cos(2.*x) + x.^3 - 14 .* x.^2 - 36 .*x + 150; %fybjtionen difineres
x = [-8 10] %der indsættes græser for funktions plottet
```

```
x = 1×2
    -8    10
```

```
fplot(f, x), grid, yline(0) %Vi plotter funktionen og tydliggør y=0
title('Den plotede function -- $f(x) = 300 \cos(2x) + x^3 - 14 x^2 - 36 x + 150$','Interpreter','latex')
xlabel('x-værdien')
ylabel('Funktionsværdien f(x)')

rod = [-4 -2 -1 1 2 4]; %Visuelt aflæste rødder skrives i en array

%laver et forloop til markering af aflæste punkter og udskriver værdien af
%disse
for i = 1:length(rod)
    xline(rod(i), '--r')
    fprintf('rod %g aflæses visuelt til %g \n', i, rod(i))
end
```

```
rod 1 aflæses visuelt til -4
rod 2 aflæses visuelt til -2
rod 3 aflæses visuelt til -1
rod 4 aflæses visuelt til 1
rod 5 aflæses visuelt til 2
```




```
rod 6 aflæses visuelt til 4
```

b)

Der skal nu foretages en søgning efter rodforekomster i et lidt større interval, nemlig fra x = -100 til 100. Dette skal gøres ved at identificere delintervaller, hvor funktionen skifter fortegn: Først undersøges det, om der er fortegnsskift fra x = -100 til -99, derefter om der er fortegnsskift fra x = -99 til -98 osv., indtil det til sidst undersøges, om der er fortegnsskift fra x = 99 til 100. Der skal altså undersøges 200 delintervaller, hvilket bekvemt kan gøre vha. MATLAB. Her følger et næsten færdigt MATLAB-script, der kan foretage undersøgelsen og undervejs udskrive delintervaller med fortegnsskift i kommandovinduet:

```
f = @(x) 300*cos(2*x)+x^3-14*x^2-36*x+150;
for x = -100:<UDFYLD>
  if f(x)*<UDFYLD> < <UDFYLD>
    disp([x,x+1])
  end
end
```

Scriptet kan downloades fra Blackboard under navnet `fortegnsskift.m`.

> Færdiggør scriptet ved at erstatte tekstangivelserne **<UDFYLD>** med korrekt MATLAB-kode. Besvarelsen skal indeholde både det færdiggjorte script og det output i kommandovinduet, som scriptet producerer, når det køres. Sammenlign resultatet med svaret i spørgsmål (a), og kommentér, hvorvidt der er overensstemmelse.

```
f = @(x) 300*cos(2*x)+x^3-14*x^2-36*x+150; %definer funktionen vi bruger

%starter forloop for at finde ud af hvornår vi krydser y=0
for x = -100:100 %vi leder i et intervald fra -100 til 100
    if f(x)*f(x+1) < 0 %hvis de to funktions værdier har ens fortegn vil der ikke ske noget
        disp([x,x+1]) % eller printes området hvor imellem der er en rod.
    end
end
```

```
    -4    -3

    -3    -2

    -2    -1

     0     1

     2     3

     3     4

    15    16
```

c)

Funktionen $f(x)$ har en rod i nærheden af $x = 2$, som kan bestemmes mere præcist vha. bisektionsmetoden. Metoden skal som udgangspunkt for søgningen bruge to startværdier, x_l og x_u. Forslag 1 er at anvende $x_l = 1$ og $x_u = 3$, og forlag 2 er at anvende $x_l = 1{,}75$ og $x_u = 2{,}25$.

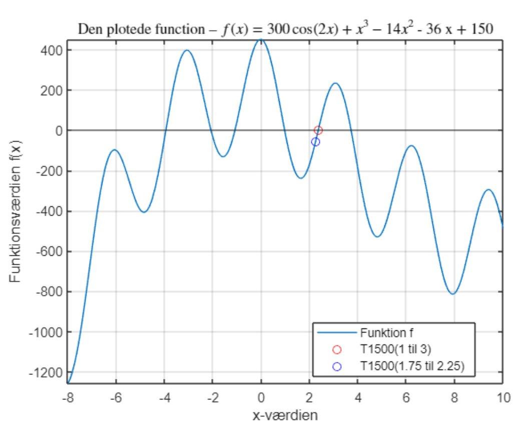
> Er begge forslag brugbare – og i så fald, er det ene forslag at foretrække fremfor det andet (begrund svaret)?

Vi undersøger først værdierne ved at bruge bisektionsmetoden, resultaterne plottes for at se hvor godt de passer

```
Tls = [1 1.75];
Tus = [3 2.25];
T1500s = [];
f = @(x) 300*cos(2*x)+x.^3-14*x.^2-36*x+150;

for i = 1:length(Tls)
    Tl = Tls(i); % Nedre intervalendepunkt (grader C)
    Tu = Tus(i); % Øvre intervalendepunkt (grader C)
    es = 0.01; % Ønsket nøjagtighed af rod (%)
    maksit = 20; % Maksimal antal iterationer
    it = 0; % Startværdi for iterationstælleren it
    while 1
        Tr = (Tl + Tu)/2; %T værdien i roden
        ea = abs(Tr - Tl)/Tr*100; %usikkerheden
        it = it + 1; %antal iterationer
        if f(Tl)*f(Tr) < 0
            Tu = Tr; %Der sættes nyt upper limit
        elseif f(Tl)*f(Tr) > 0
            Tl = Tr; %Der sættes nyt lower limit
        else
            ea = 0; %usikkerheden er 0 da vi har ramt rigtigt
        end
        %Forloopet brydes hvis vi når et maksimalt antal forsøg eller vi kommer
        %indenfor usikkerheden.
        if ea <= es || it >= maksit
            break
        end
    end
    T1500 = Tr % Temperaturen printes
    T1500s(end + 1) = T1500;

    ea % Usikkerheden printes
    it % Antal interationer der er gennemgået
if i == length(Tls)
    break
end
    disp('---------------------------------------')
end
```

```
T1500 = 2.3551
ea = 0.0052
it = 14
--------------------------------------
T1500 = 2.2499
ea = 0.0054
it = 12
```

```
x = [-8 10];
f = @(x) 300 .* cos(2.*x) + x.^3 - 14 .* x.^2 - 36 .*x + 150;
fplot(f, x), grid, yline(0); %Vi plotter funktionen og tydliggør y=0
hold on

T1 = plot(T1500s(1), f(T1500s(1)), 'or','DisplayName', 'T1500(1 til 3)');
T2 = plot(T1500s(2), f(T1500s(2)), 'ob','DisplayName', 'T1500(1.75 til 2.25)');

legend('Funktion f','', 'T1500(1 til 3)', 'T1500(1.75 til 2.25)', Location='best')
title('Den plotede function -- $f(x) = 300 \cos(2x) + x^3 - 14 x^2$ - 36 x + 150','Interpreter','latex')
xlabel('x-værdien')
ylabel('Funktionsværdien f(x)')
hold off
```



Det ses på plottet at værdierne 1 og 3 er best egnede til at finde roden. Det bemærkes således at 1.75 til 2.25 giver resultatet der er grænseværdien. Når vi opnår grænseværdien skal vi være skeptiske og prøve at udvide vores grænser.

Vi kan konkludere at jo mindre interval vi har jo hurtiger køre koden, men her er der også størrer risiko for ikke at have nul punktet med i intervallet. omvendt må vores interval ikke blive for stort da det så kan komme til at inkludere to eller flere 0 punkter.