

UWTSD

MSc Motorsport Engineering

Task 2

ACHA7002 - Alternative Fuels Combustion Modelling

2500151

5th December 2025

Contents

1	Bisection	1
1.1	$fx = \sin x \sinh(x)$	1
1.2	$fx = R^2 - x^{21/2} - x \tan(x)$	1
2	Newton's method	2
2.1	Newton b $fx = ex - (1 + x + (x^2/2))$	3
2.2	Newton c $fx = 10 \log x - x$	4
3	Intersect plot $fx = 100e - x - 5 \sin(\pi x/2)$	5
4	Bracketing	6
4.1	Bisect $fx = 100e - x - 5 \sin(\pi x/2)$	6
4.2	Newton $fx = 100e - x - 5 \sin(\pi x/2)$	7
5	Appendices	7
5.1	Convergence table Bisection $fx = \sin x \sinh(x)$	7
5.2	Convergence table Bisection $fx = R^2 - x^{21/2} - x \tan(x)$	8
5.3	Convergence table Newton b $fx = ex - (1 + x + (x^2/2))$	9
5.4	Convergence table Newton c $fx = 10 \log x - x$	9
5.5	Convergence table Bisect $fx = 100e - x - 5 \sin(\pi x/2)$	10
5.6	Convergence table Newton $fx = 100e - x - 5 \sin(\pi x/2)$	12
5.7	Your code and functions	13

1 Bisection

A function was created based on equation 1.1 to solve roots using the bisection method

$$x_2 = (x_1 + x_3)/2 \quad \text{Equation 1-1}$$

$$1.1 \quad f(x) = \sin(x) \sinh(x)$$

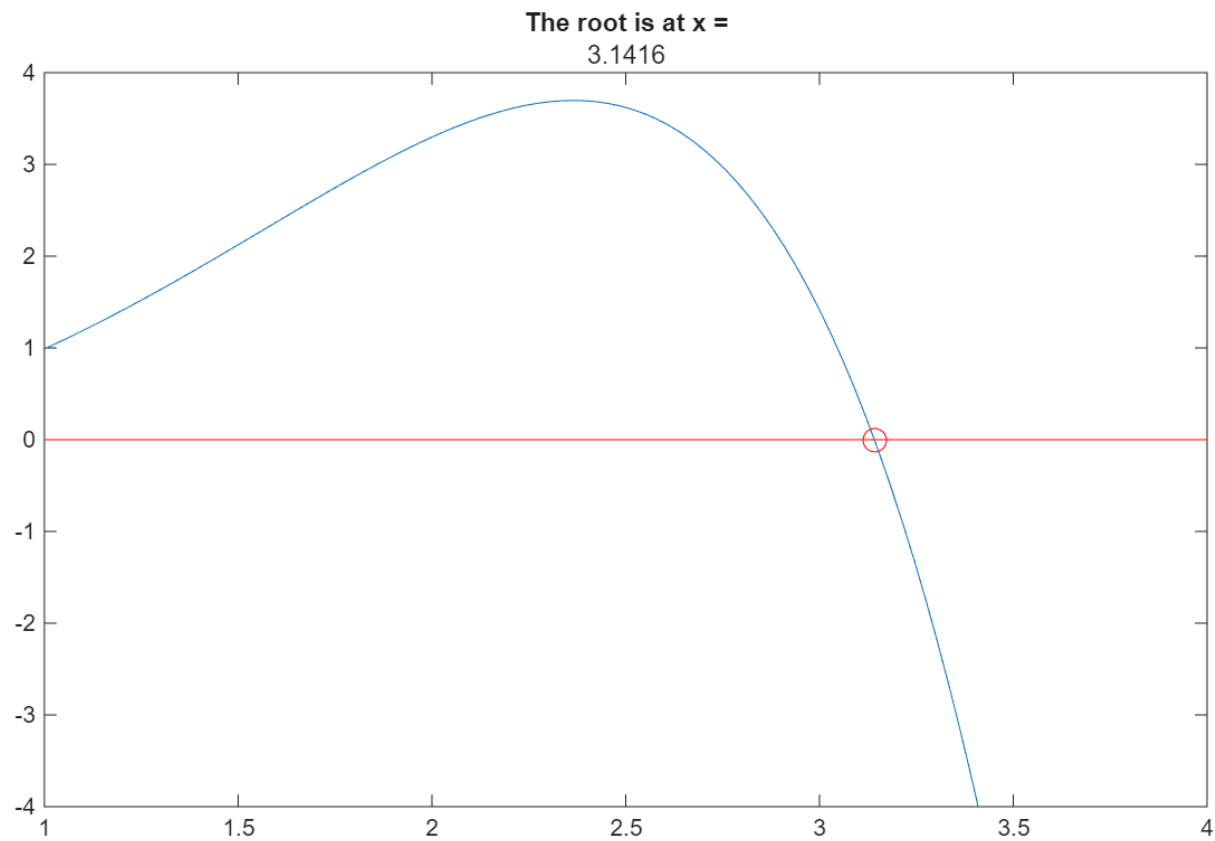


Figure 1-1 this shows the function labelled in the legend and the root

$$1.2 \quad f(x) = (R^2 - x^2)^{1/2} - x \tan(x)$$

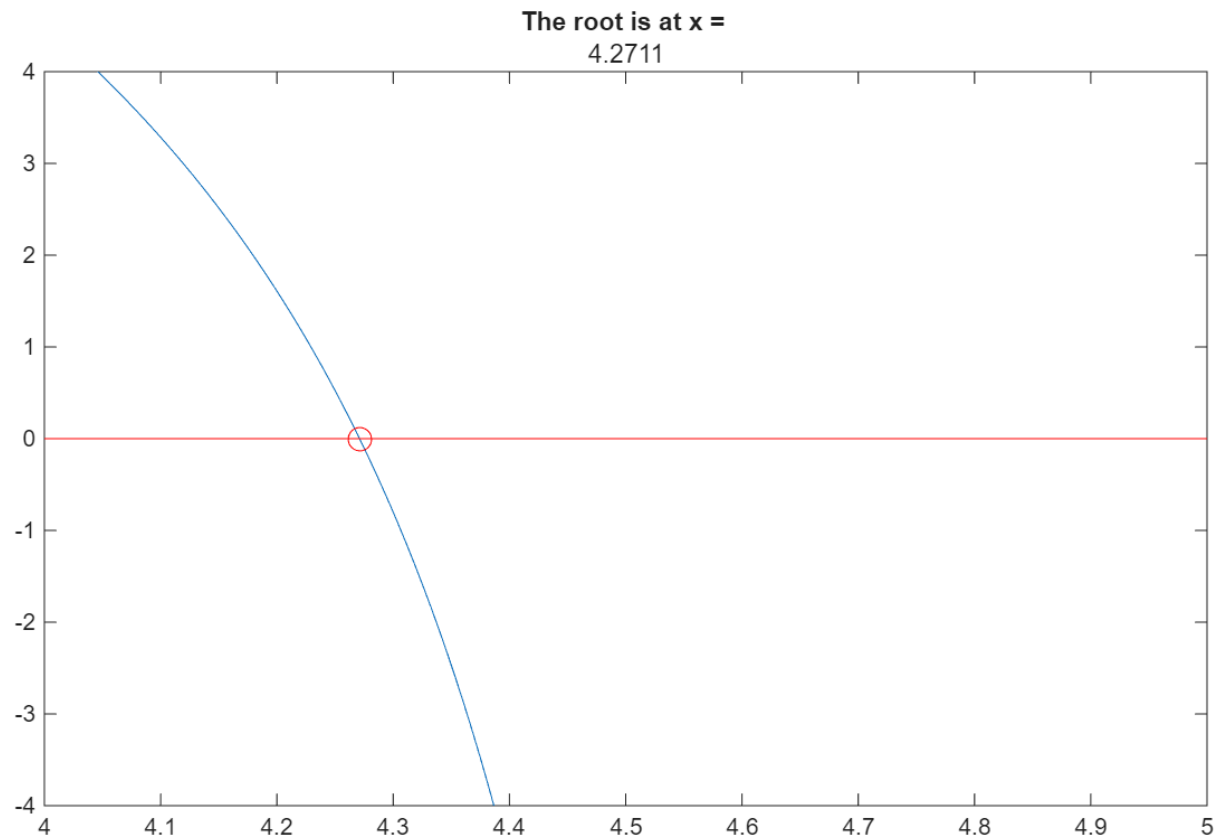


Figure 1-2 this shows the function labelled in the legend and the root

2 Newton's method

A function was made from equation 2.1 to evaluate roots with the newton method to use this method, functions and their derivative functions were created.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Equation 2-1

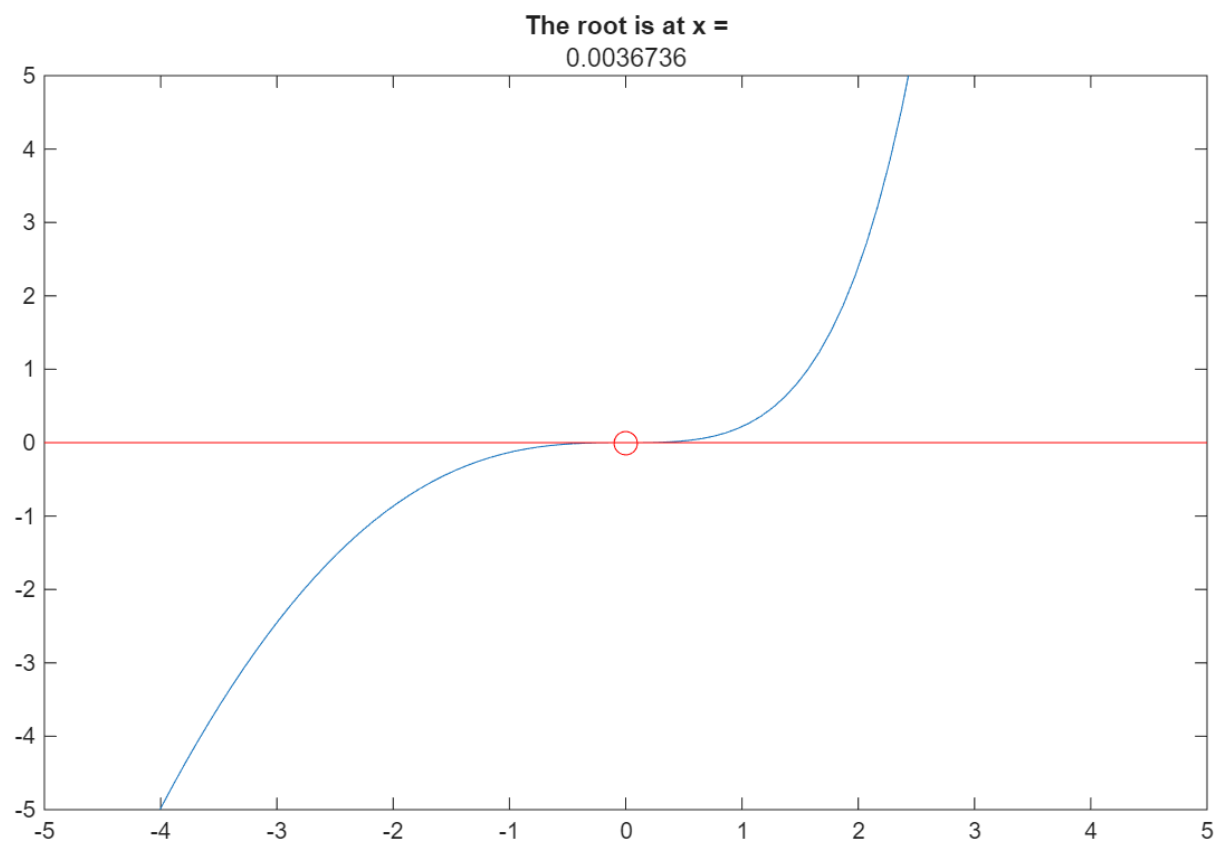
2.1 Newton b $f(x) = e^x - (1 + x + (x^2/2))$ 

Figure 2-1 this shows the function labelled in the legend and the root

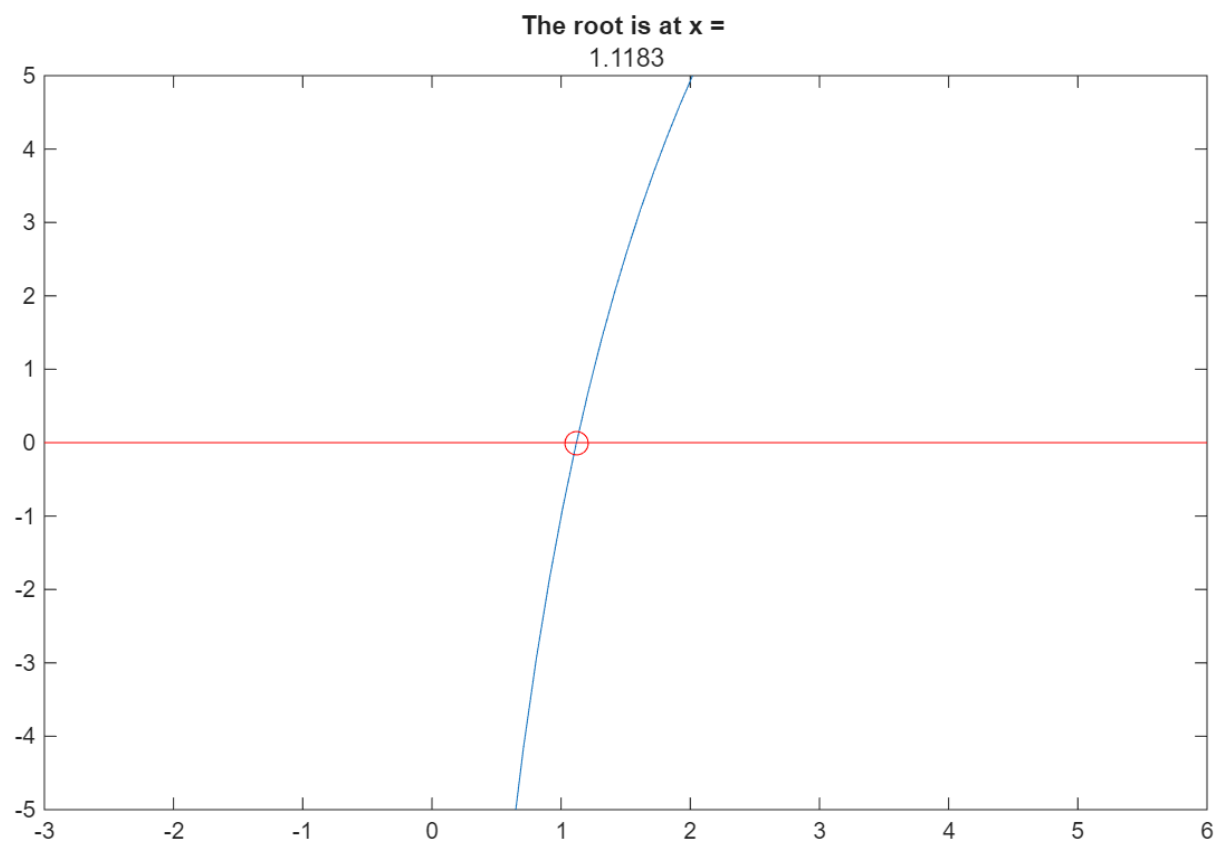
2.2 Newton c $f(x) = 10 \log(x) - x$ 

Figure 2-2 this shows the function labelled in the legend and the root

3 Intersect plot $f(x) = 100e^{-x} - 5\sin(\pi x/2)$

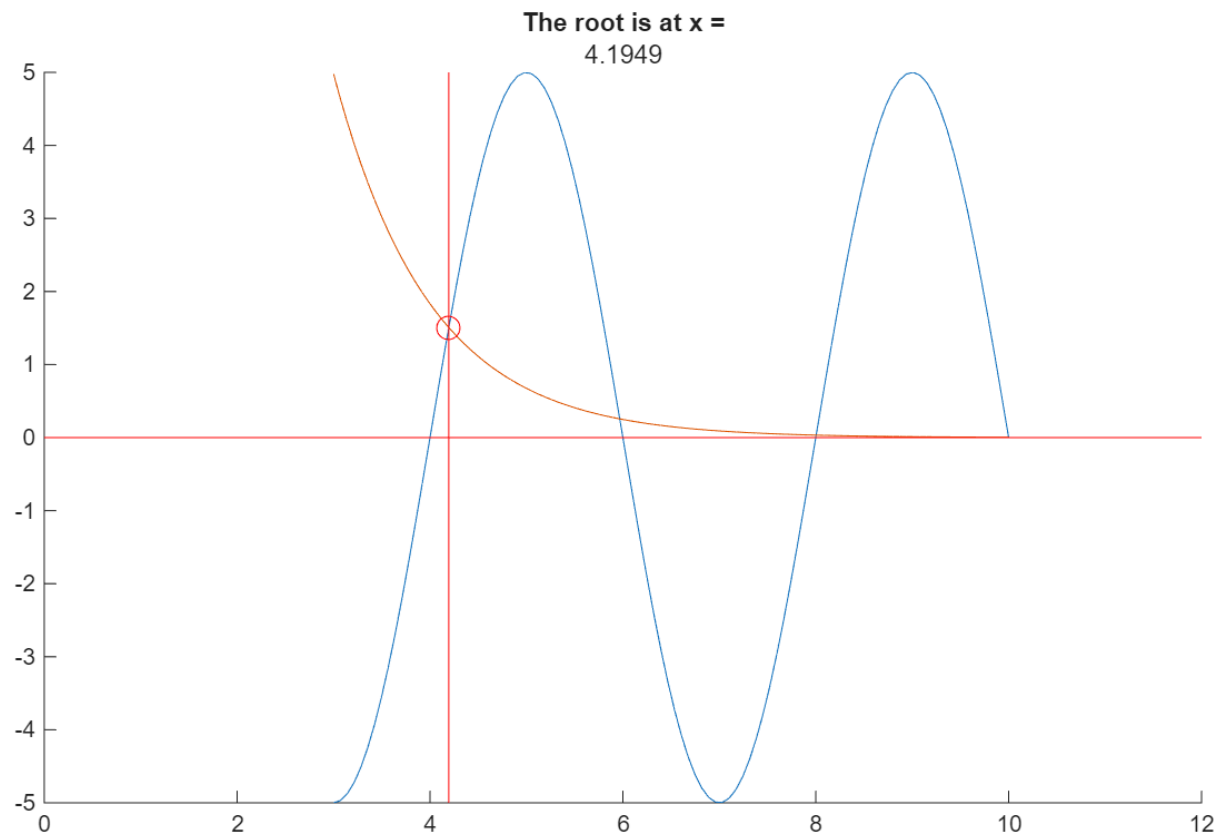


Figure 3-1 plot of function intersection

4 Bracketing

4.1 Bisect $f(x) = 100e^{-x} - 5\sin(\pi x/2)$

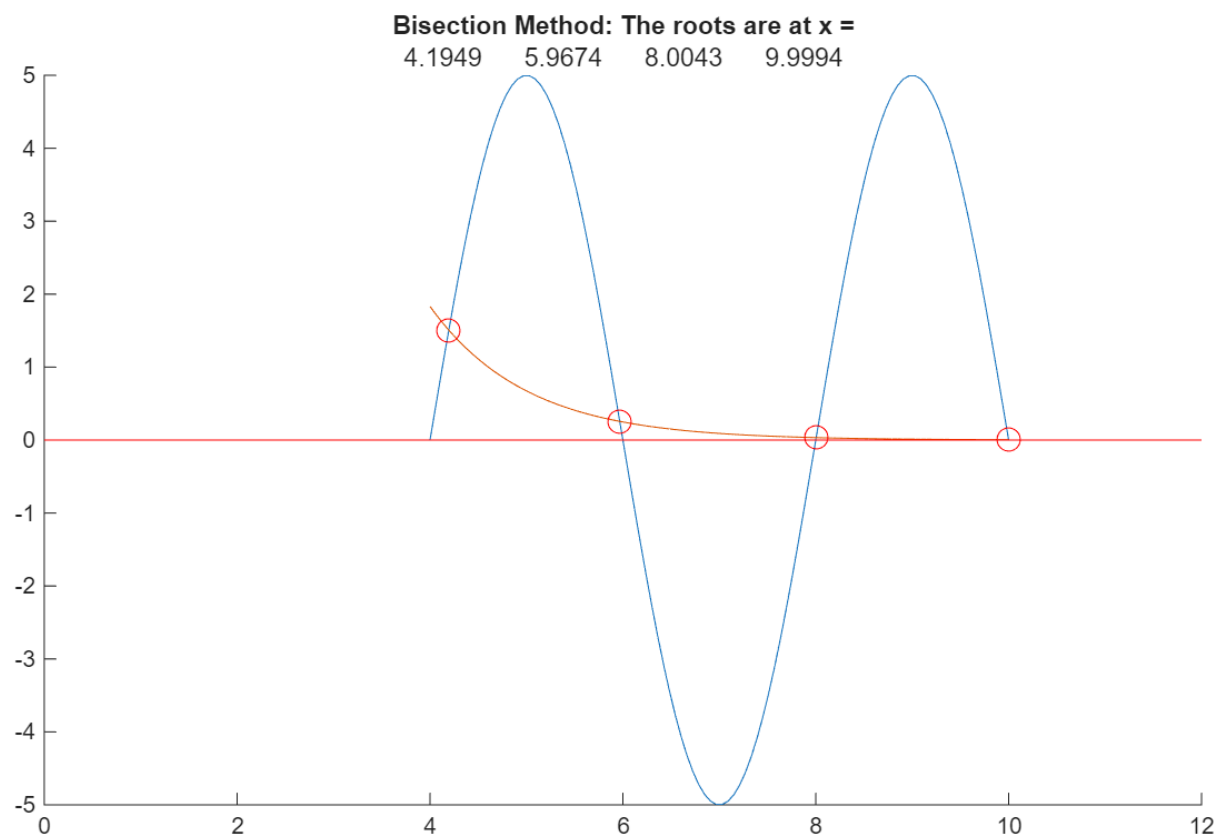


Figure 4-1 multiple roots Bisect for functions shown in legend

4.2 Newton $f(x) = 100e^{-x} - 5\sin(\pi x/2)$

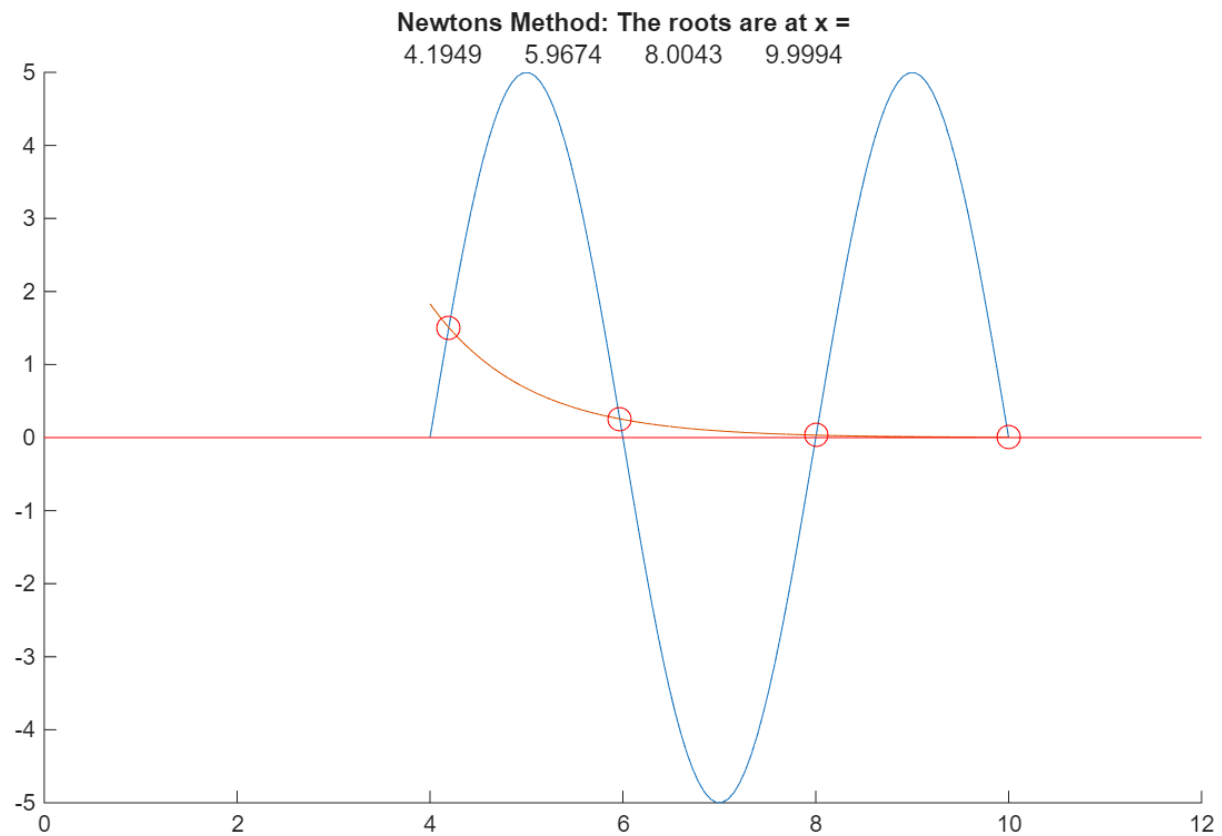


Figure 4-2 multiple roots Newton for functions shown in legend

5 Appendices

5.1 Convergence table Bisection $f(x) = \sin(x) \sinh(x)$

Iterations	Lower Guess	Upper Guess	Mid-Point
1	1.000000	4.000000	2.500000
2	2.500000	4.000000	3.250000
3	2.500000	3.250000	2.875000
4	2.875000	3.250000	3.062500
5	3.062500	3.250000	3.156250
6	3.062500	3.156250	3.109375
7	3.109375	3.156250	3.132812
8	3.132812	3.156250	3.144531
9	3.132812	3.144531	3.138672
10	3.138672	3.144531	3.141602

11	3.138672	3.141602	3.140137
12	3.140137	3.141602	3.140869
13	3.140869	3.141602	3.141235
14	3.141235	3.141602	3.141418
15	3.141418	3.141602	3.141510
16	3.141510	3.141602	3.141556
17	3.141556	3.141602	3.141579
18	3.141579	3.141602	3.141590
19	3.141590	3.141602	3.141596
20	3.141590	3.141596	3.141593
21	3.141590	3.141593	3.141592
22	3.141592	3.141593	3.141592

5.2 Convergence table Bisection $f(x) = (R^2 - x^2)^{1/2} - x \tan(x)$

Iterations	Lower Guess	Upper Guess	Mid-Point
1	4.000000	4.700000	4.350000
2	4.000000	4.350000	4.175000
3	4.175000	4.350000	4.262500
4	4.262500	4.350000	4.306250
5	4.262500	4.306250	4.284375
6	4.262500	4.284375	4.273437
7	4.262500	4.273437	4.267969
8	4.267969	4.273437	4.270703
9	4.270703	4.273437	4.272070
10	4.270703	4.272070	4.271387
11	4.270703	4.271387	4.271045
12	4.271045	4.271387	4.271216
13	4.271045	4.271216	4.271130
14	4.271045	4.271130	4.271088
15	4.271088	4.271130	4.271109
16	4.271088	4.271109	4.271098
17	4.271088	4.271098	4.271093
18	4.271093	4.271098	4.271096
19	4.271093	4.271096	4.271094

20	4.271094	4.271096	4.271095
----	----------	----------	----------

5.3 Convergence table Newton b $f(x) = e^x - (1 + x + (x^2/2))$

Iterations	First Guess	Next Guess	Absolute Error
1	5.000000	4.087773	0.912227
2	4.087773	3.241020	0.846752
3	3.241020	2.487381	0.753639
4	2.487381	1.849522	0.637859
5	1.849522	1.337187	0.512335
6	1.337187	0.944907	0.39228
7	0.944907	0.656151	0.288756
8	0.656151	0.449883	0.206268
9	0.449883	0.305706	0.144177
10	0.305706	0.206451	0.0992545
11	0.206451	0.138834	0.0676172
12	0.138834	0.093096	0.0457378
13	0.093096	0.062307	0.0307899
14	0.062307	0.041646	0.0206606
15	0.041646	0.027812	0.0138337
16	0.027812	0.018563	0.00924924
17	0.018563	0.012385	0.0061781
18	0.012385	0.008261	0.00412405
19	0.008261	0.005509	0.00275174
20	0.005509	0.003674	0.00183554

Maximum number of iterations reached without convergence.

5.4 Convergence table Newton c $f(x) = 10 \log(x) - x$

Iterations	First Guess	Next Guess	Absolute Error
1	1.000000	1.111111	0.111111
2	1.111111	1.118299	0.00718824
3	1.118299	1.118326	2.62358e-05
4	1.118326	1.118326	3.46505e-10

5.5 Convergence table Bisect $f(x) = 100e^{-x} - 5\sin(\pi x/2)$

Iterations	Lower Guess	Upper Guess	Mid-Point
1	4.000000	5.000000	4.500000
2	4.000000	4.500000	4.250000
3	4.000000	4.250000	4.125000
4	4.125000	4.250000	4.187500
5	4.187500	4.250000	4.218750
6	4.187500	4.218750	4.203125
7	4.187500	4.203125	4.195312
8	4.187500	4.195312	4.191406
9	4.191406	4.195312	4.193359
10	4.193359	4.195312	4.194336
11	4.194336	4.195312	4.194824
12	4.194824	4.195312	4.195068
13	4.194824	4.195068	4.194946
14	4.194824	4.194946	4.194885
15	4.194885	4.194946	4.194916
16	4.194916	4.194946	4.194931
17	4.194931	4.194946	4.194939
18	4.194931	4.194939	4.194935
19	4.194931	4.194935	4.194933
20	4.194931	4.194933	4.194932

Iterations	Lower Guess	Upper Guess	Mid-Point
1	5.500000	6.500000	6.000000
2	5.500000	6.000000	5.750000
3	5.750000	6.000000	5.875000
4	5.875000	6.000000	5.937500
5	5.937500	6.000000	5.968750
6	5.937500	5.968750	5.953125
7	5.953125	5.968750	5.960938
8	5.960938	5.968750	5.964844
9	5.964844	5.968750	5.966797

10	5.966797	5.968750	5.967773
11	5.966797	5.967773	5.967285
12	5.967285	5.967773	5.967529
13	5.967285	5.967529	5.967407
14	5.967285	5.967407	5.967346
15	5.967346	5.967407	5.967377
16	5.967377	5.967407	5.967392
17	5.967377	5.967392	5.967384
18	5.967377	5.967384	5.967381
19	5.967377	5.967381	5.967379
20	5.967379	5.967381	5.967380

Iterations	Lower Guess	Upper Guess	Mid-Point
1	7.500000	8.000000	8.000000
2	8.000000	8.500000	8.250000
3	8.000000	8.250000	8.125000
4	8.000000	8.125000	8.062500
5	8.000000	8.062500	8.031250
6	8.000000	8.031250	8.015625
7	8.000000	8.015625	8.007812
8	8.000000	8.007812	8.003906
9	8.003906	8.007812	8.005859
10	8.003906	8.005859	8.004883
11	8.003906	8.004883	8.004395
12	8.003906	8.004395	8.004150
13	8.004150	8.004395	8.004272
14	8.004150	8.004272	8.004211
15	8.004211	8.004272	8.004242
16	8.004242	8.004272	8.004257
17	8.004242	8.004257	8.004250
18	8.004250	8.004257	8.004253
19	8.004250	8.004253	8.004251
20	8.004251	8.004253	8.004252

Iterations	Lower Guess	Upper Guess	Mid-Point
1	9.000000	10.000000	9.500000
2	9.500000	10.000000	9.750000
3	9.750000	10.000000	9.875000
4	9.875000	10.000000	9.937500
5	9.937500	10.000000	9.968750
6	9.968750	10.000000	9.984375
7	9.984375	10.000000	9.992188
8	9.992188	10.000000	9.996094
9	9.996094	10.000000	9.998047
10	9.998047	10.000000	9.999023
11	9.999023	10.000000	9.999512
12	9.999023	9.999512	9.999268
13	9.999268	9.999512	9.999390
14	9.999390	9.999512	9.999451
15	9.999390	9.999451	9.999420
16	9.999420	9.999451	9.999435
17	9.999420	9.999435	9.999428
18	9.999420	9.999428	9.999424
19	9.999420	9.999424	9.999422
20	9.999420	9.999422	9.999421

5.6 Convergence table Newton $f(x) = 100e^{-x} - 5\sin(\pi x/2)$

Iterations	First Guess	Next Guess	Absolute Error
1	4.000000	4.189103	0.189103
2	4.189103	4.194922	0.00581905
3	4.194922	4.194932	9.71121e-06
4	4.194932	4.194932	2.73923e-11

Iterations	First Guess	Next Guess	Absolute Error
1	5.500000	6.107755	0.607755

2	6.107755	5.966141	0.141615
3	5.966141	5.967379	0.00123814
4	5.967379	5.967379	9.1339e-08

Iterations	First Guess	Next Guess	Absolute Error
1	7.500000	8.140203	0.640203
2	8.140203	8.002019	0.138184
3	8.002019	8.004253	0.00223458
4	8.004253	8.004253	2.75412e-08

Iterations	First Guess	Next Guess	Absolute Error
1	9.500000	10.136129	0.636129
2	10.136129	9.997299	0.138831
3	9.997299	9.999422	0.00212299
4	9.999422	9.999422	1.23981e-08

5.7 Your code and functions

```
% Root solving 1. Bisection Method

% fprintf('Final Estimate of the Root = %10.6f \r \r',mid_point)
x1 = [1, 4];
x2 = [4, 4.7];
x_1 = linspace(1, 4, 100)';
x_2 = linspace (4, 4.7, 100)';

R = 10;
imax = 50;
tol = 1e-6;

f_of_x1 = sinsinh(x_1);
f_of_x2 = squaretoot (x_2);

root_1 = bisection(@sinsinh, 1, 4, imax, tol);
root_2 = bisection (@squaretoot, 4, 4.7, imax, tol);

fprintf('Root found in interval [1, 4]: %10.6f\n', root_1);
fprintf('Root found in interval [4, 4.7]: %10.6f\n \r', root_2);

% plot the first function
figure (1)
plot(x_1, f_of_x1)
hold on
plot ([1 4], [0 0], 'r') % plot y=0 axis
plot (root_1, 0, 'ro', 'MarkerSize',10)
```

```

title('The root is at x =', num2str(root_1))
hold off
ylim ([-4 4])

% Plot the second function
figure(2)
plot(x_2, f_of_x2)
hold on
plot([4 5], [0 0], 'r') % plot y=0 axis
plot (root_2, 0, 'ro', 'MarkerSize',10)
title('The root is at x = ',num2str(root_2))
hold off
ylim([-4 4])

%% Root solving 2. Newton's Method

N_1 = linspace(-5, 5, 100)';
N_2 = linspace (0, 10, 100)';

f_of_N1 = esp(N_1);
f_of_N2 = logaritmo(N_2);

tol_S = 1e-6;
imax_S = 20;

root_S1 = newton(5, @esp, @d_esp,imax_S, tol_S);
root_S2 = newton(1, @logaritmo, @d_logaritmo, imax_S, tol_S);

% plot the first function
figure (3)
plot(N_1, f_of_N1)
hold on
plot ([-5 5], [0 0], 'r') % plot y=0 axis
plot (root_S1, 0, 'ro', 'MarkerSize',10)
title('The root is at x =', num2str(root_S1))
hold off
ylim ([-5 5])

% Plot the second function
figure(4)
plot(N_2, f_of_N2)
hold on
plot([-3 6], [0 0], 'r') % plot y=0 axis
plot (root_S2, 0, 'ro', 'MarkerSize',10)
title('The root is at x = ',num2str(root_S2))
hold off
ylim([-5 5])

%% Root solving 3. Newton's Method - Intersection of two functions

t = linspace(3, 10, 100);

f_of_t1 = seq(t);
f_of_t2 = teq(t);

tol_3 = 1e-6;
imax_3 = 20;

```

```

root_t = newton(4, @peq,@d_peq, imax_3, tol_3);
f_of_root = seq (root_t);

figure (5)
hold on
plot(t, f_of_t1)
plot(t, f_of_t2)

plot ([0 12], [0 0], 'r') % plot y=0 axis
plot (root_t, f_of_root, 'ro', 'MarkerSize',10)
plot ([root_t root_t], [-5 5], 'r') % plot x = root axis

title('The root is at x =', num2str(root_t))
hold off
ylim ([-5 5])

% same thing but with tol of 1e-10
tol_3b = 1e-10;
root_tb = newton(4, @peq,@d_peq, imax_3, tol_3b);

%Only 4 iterations are necessary to achieve the tolerance of 1e-10 since
%the Newton's Method converges very quickly

%% Root solving 4. Bracketing (Multiple Roots)

u = linspace(4, 10, 100);

f_of_u1 = seq(u);
f_of_u2 = teq(u);

tol4 = 1e-6;
imax4 = 30;

%I'm lazy, so I will just plot the equation and check with my eyes where
%the approximate roots are

figure (6)
hold on
plot(u, f_of_u1)
plot(u, f_of_u2)

plot ([4 11], [0 0], 'r') % plot y=0 axis
title('Checking where the approximate roots are')
hold off

%Roots are approximately at x = 4, 6, 8, 10

r1_b = bisect(@peq, 4, 5, imax4, tol4);
r2_b = bisect(@peq, 5.5, 6.5, imax4, tol4);
r3_b = bisect(@peq, 7.5, 8.5, imax4, tol4);
r4_b = bisect(@peq, 9, 10, imax4, tol4);

fprintf('Roots found in intervals [4, 5] with Bisection Method: %10.6f\n', r1_b);
fprintf('Roots found in intervals [5.5, 6.5] with Bisection Method: %10.6f\n',
r2_b);
fprintf('Roots found in intervals [7.5, 8.5] with Bisection Method: %10.6f\n',
r3_b);
fprintf('Roots found in intervals [9, 10] with Bisection Method: %10.6f\n', r4_b);

```

```

f_of_r1_b = seq (r1_b);
f_of_r2_b = seq (r2_b);
f_of_r3_b = seq (r3_b);
f_of_r4_b = seq (r4_b);

figure (7)
hold on
plot(u, f_of_u1)
plot(u, f_of_u2)

plot ([0 12], [0 0], 'r') % plot y=0 axis
plot (r1_b, f_of_r1_b, 'ro', 'MarkerSize',10)
plot (r2_b, f_of_r2_b, 'ro', 'MarkerSize',10)
plot (r3_b, f_of_r3_b, 'ro', 'MarkerSize',10)
plot (r4_b, f_of_r4_b, 'ro', 'MarkerSize',10)

roots_b = [r1_b r2_b r3_b r4_b];

title('Bisection Method: The roots are at x =', num2str(roots_b))
hold off
ylim ([-5 5])

r1_t = newton(4, @peq,@d_peq, imax4, tol4);
r2_t = newton(5.5, @peq,@d_peq, imax4, tol4);
r3_t = newton(7.5, @peq,@d_peq, imax4, tol4);
r4_t = newton(9.5, @peq,@d_peq, imax4, tol4);

f_of_r1_t = seq (r1_t);
f_of_r2_t = seq (r2_t);
f_of_r3_t = seq (r3_t);
f_of_r4_t = seq (r4_t);

fprintf('Roots found in intervals [4, 5] with Newtons Method: %10.6f\n', r1_t);
fprintf('Roots found in intervals [5.5, 6.5] with Newtons Method: %10.6f\n',
r2_t);
fprintf('Roots found in intervals [7.5, 8.5] with Newtons Method: %10.6f\n',
r3_t);
fprintf('Roots found in intervals [9, 10] with Newtons Method: %10.6f\n', r4_t);

figure (8)
hold on
plot(u, f_of_u1)
plot(u, f_of_u2)

plot ([0 12], [0 0], 'r') % plot y=0 axis
plot (r1_t, f_of_r1_t, 'ro', 'MarkerSize',10)
plot (r2_t, f_of_r2_t, 'ro', 'MarkerSize',10)
plot (r3_t, f_of_r3_t, 'ro', 'MarkerSize',10)
plot (r4_t, f_of_r4_t, 'ro', 'MarkerSize',10)

roots_t = [r1_t r2_t r3_t r4_t];

title('Newtons Method: The roots are at x =', num2str(roots_t))
hold off
ylim ([-5 5])

```

```
function ss = sinsinh (x)

    ss = sin(x).*sinh(x);

end
```

```
function s = squaretoot (x)
R = 10;

s = sqrt(R.^2-x.^2) - x.*tan(x);

end
```

```
function [r] = bisect(fun,a,b,imax,tol)
% input: f = function to be evaluated
% a, b = interval limits
% imax = iteration maximum
% tol = convergence criterion
% output: r = position of the root

% initialise parameter, i.e.
x1 = a;
f1 = feval(fun, x1);
x3 = b;
f3 = feval(fun, x3);

% check: verify that there is indeed a root in the interval [a, b].
if f1*f3 > 0
    error ('No root n the specified interval')
end

% Begin iteration loop i.e.
counter = 1;

abs_diff= inf;

while (abs(abs_diff)>tol && counter <imax)
    f1 = feval(fun, x1);
    f3 = feval(fun, x3);

    % calculate mid-point
    x2 = (x1+x3)/2;
    f2 = feval (fun, x2);

    % output to screen
    fprintf('Bisection Method: ITERATION %i \r', counter)
    fprintf('Lower guess = %10.6f \r', x1)
    fprintf('Upper guess = %10.6f \r', x3)
    fprintf('Mid-point = %10.6f \r \r', x2)

    % decision
    if f1*f2<0
        %Root is in left half, so
        r = (x2+x1)/2;
```

```

        f3=f2;
        x3=x2;

    else
        %root is in right half, so
        r = (x2+x3)/2;
        f1=f2;
        x1=x2;
    end
    %to test tolerance
    abs_diff = abs(x3 - x1); % Update the absolute difference for convergence check

    counter = counter + 1;
end

end

```

```

function esp = esp(x)

esp = exp(x)-(1+x+(x.^2)/2);

end

```

```

function d_esp = d_esp (x)

d_esp = exp(x) -1 -x;

end

```

```

function l = logaritmo (x)

l = 10*log(x)-x;

end

```

```

function d_l = d_logaritmo(x)

d_l = 10./(x) - 1;

end

```

```

function [root] = newton (x, fun, df, imax, tol)

% input:    x = initial guess at the root
% f = function to be evaluated
% df = derivative of f with respect to x
% imax = iteration maximum
% tol = convergence criterion
% output: root = position of the root

xold = x;

```

```

fold = feval (fun, xold);
dfold = feval(df, xold);

% CHECK: verify that the derivative is not zero
count = 1;

%Set next guess to infinity to fool while condition
abs_diff_between_guesses = inf;

% start iterating - note stop conditions
while (abs_diff_between_guesses>tol) && (count<=imax)
    % if df==0
    if isempty(df)
        error('Derivative is undefined at this point.')
    end

    % Update function values for the next iteration
    fold = feval(fun, xold);
    dfold = feval(df, xold);

    %Calculate next point using secant formula
    x_next = xold - fold/dfold;

    % output to screen
    fprintf('Newtons Method: ITERATION %i \r', count)
    fprintf ('First Guess = %10.6f \r', xold)
    fprintf ('Next Guess = %10.6f \r \r',x_next)

    %To test tolerance
    abs_diff_between_guesses = abs(x_next - xold);

    %Update the guesses
    xold = x_next;

    root = xold;

    fprintf('Iteration = %i,\t Error = %g\n
\r',count,norm(abs_diff_between_guesses))

    % Increment the iteration count
    count = count + 1;

end

if abs_diff_between_guesses>tol
    fprintf('Maximum number of iterations reached without convergence.\r \r');
end

end

function peq = peq(x)

peq = 100*exp(-x)-5*sin((pi*x)/2);

end

```

```
function seq = seq(x)
seq = 5*sin((pi*x)/2);
end
```

```
function teq = teq(x)
teq = 100*exp(-x);
end
```

```
function d = d_peq(x)
d = -(200+5*pi*exp(x)*cos((pi*x)/2))/(2*exp(x));
end
```