Department of Engineering Mathematics

EMAT20920: Numerical Methods in MATLAB

COURSEWORK ASSESSMENT

Jake Bowhay (UP19056)

All figures in this report have been saved using `saveFigPDF` function as it automatically resizes the paper to the correct size.

```
function saveFigPDF(fileName)
    % saveFigPDF saves open figure as a PDF file
    %
    % Inputs:
    %   fileName = File name to save figure as
    % Usage:
    %   saveFigPDF("polynomial") -> Saves current figure as polynomial.pdf

    % Get current figure handle
    figureHandle = gcf;
    % Resize paper
    set(figureHandle,'PaperPosition',3*[0 0 6 4]);
    set(figureHandle,'PaperSize',3*[6 4]);
    set(figureHandle,'PaperUnits','centimeters');

    print(fileName,'-dpdf');
end
```

Question 1: Root-finding

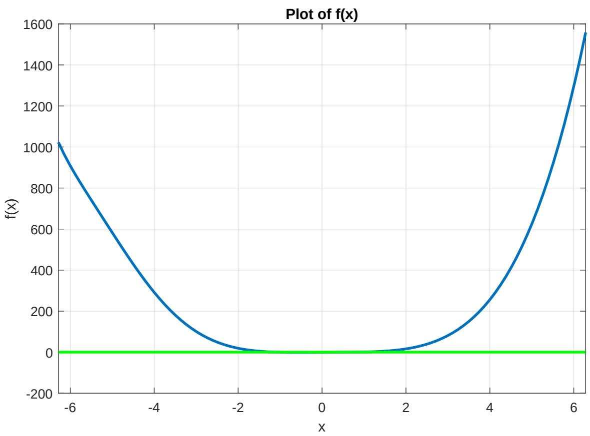
(a) To find how many solutions each equation has in the given domain I will rearrange all the equations to be equal to zero and then looks for the zeros of the rearranged equations. As a corollary to the intermediate value theorem, if a function is continuous and changes sign in a bracket then that bracket must contain a zero. So I will plot each of the rearranged equation and I look for appropriate brackets. I will use the `pltFunc` function to plot the functions as it removes values outside a defined limit which prevents MATLAB plotting discontinuous functions as continuous. The limits can then be changed using the property explorer to give a more useful plot.

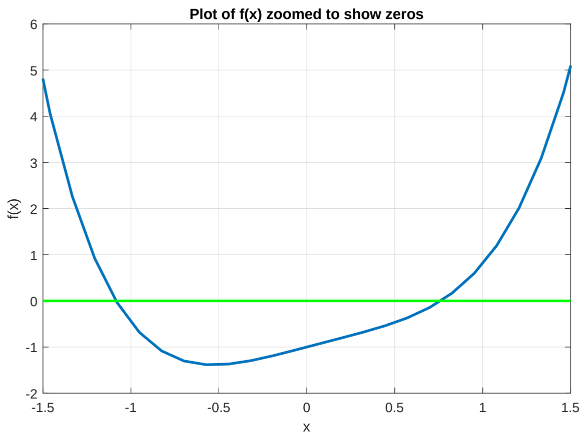
```
function pltFunc(f, domain, discontLim)
    %pltFunc plots function f between values of xLim removing any values
    % that are greater than discontLim to prevent MATLAB plotting
    % discontinuous functions as continous and plots a line of x = 0 to
    % help make any zeros clear
    %
    % Input:
    %   f = function handle to plot
    %   domain = 1x2 vector containing the lower and upper bound of the
    %   domain of f
    %   discountLim = absolute values of the function greater than this are
    %   changed to NaN. Setting to inf will plot all values of the function
    %
    % Usage:
    %   pltFunc(@(x) 1./x, [-10 10], 5) -> Plots 1/x between -10 and 10
    %   changing the values where |1/x|>5 to NaN
```

```
    % Check xLim is the correct dimensions
    assert(isequal(size(domain), [1 2]), "xLim must be a 1x2 vector")

    %% Generate values to plot
    x = linspace(domain(1), domain(2));
    y = f(x);

    % Remove large values of y to prevent MATLAB plotting discontinuous
    % functions as continuous
    y(abs(y)>discontLim) = NaN;

    %% Plot function and line x = 0
    plot(x, y, [min(x) max(x)], [0 0], "g-", "LineWidth", 2);
    xlabel("x");
    ylabel("f(x)");
    xlim(domain);
    title("Plot of f(x)");
    grid on;
end
```

(i) Rearranging $x^4 = e^{-x}\cos(x)$ gives $f(x) = x^4 - e^{-x}\cos(x)$.

```
f = @(x) x.^4 - exp(-x).*cos(x);
pltFunc(f, [-2*pi 2*pi], inf);
```

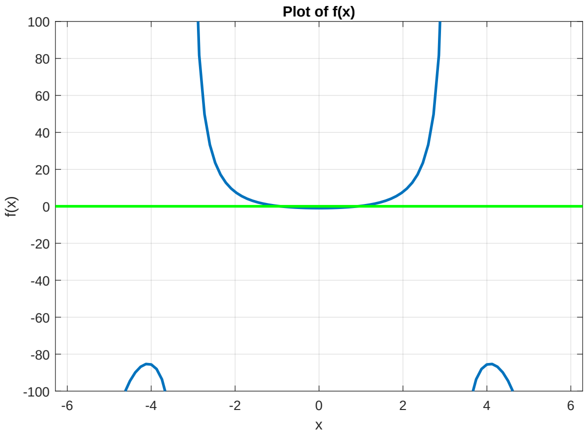


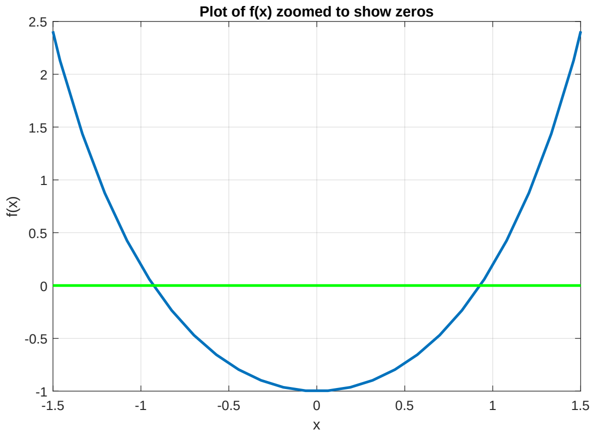


The second zoomed in plot shows there are two zeros in the given domain. The first zero can be bracketed by the interval $[-1.5, -1]$ as $f(-1.5) = 4.7455$ and $f(-1) = -0.4687$ so since the function is continuous and there is a change of sign this bracket must contain a zero. Like wise the second root can be bracketed by the interval $[0.5, 1]$ as $f(0.5) = -0.4698$ and $f(1) = 0.8012$.

(ii) Setting $f(x) = \frac{x^3}{\sin(x)} - 1$.

```
f = @(x) (x.^3)./sin(x) - 1;
pltFunc(f, [-2*pi 2*pi], 500);
```

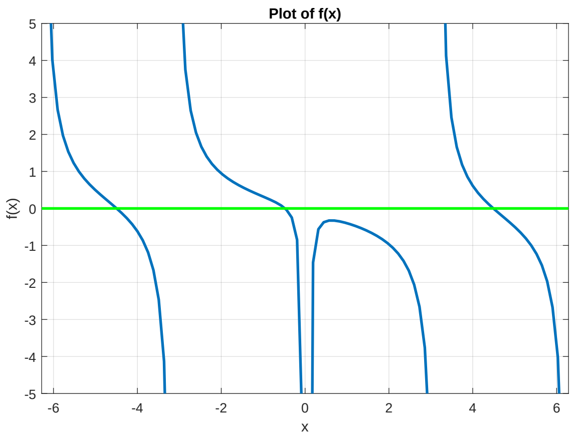




The second plot show there are two roots. The first root can be bracketed by the interval $[-1, -0.5]$ as $f(-1) = 0.1884$ and $f(-0.5) = -0.7393$ and $f(x)$ is continuous in this bracket. Likewise, the second root can be bracketed by the interval $[0.5, 1]$ as $f(0.5) = -0.7393$ and $f(1) = 0.1884$.

(iii) Rearranging $\cot(x) = \frac{25}{25x-1}$ gives $f(x) = \cot(x) - \frac{25}{25x-1}$.

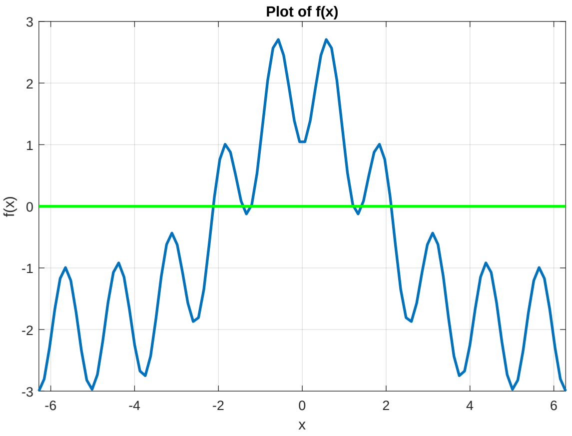
```
f = @(x) cot(x) - 25./(25*x - 1);
pltFunc(f, [-2*pi 2*pi], 30);
```

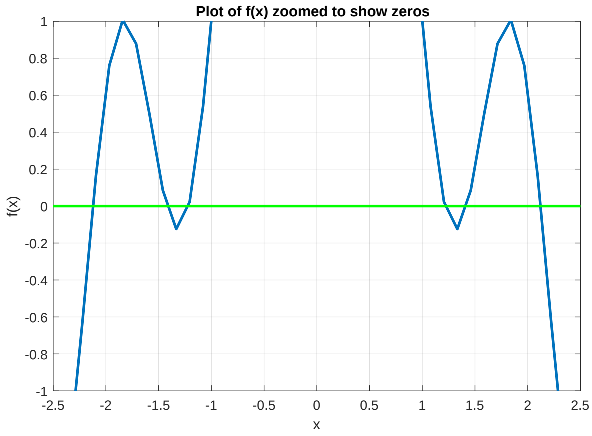


The plot shows that the equation has three solutions. The first can be bracketed by the interval $[-5, -4]$ as $f(-5) = 0.4942$ and $f(-4) = -0.6162$. The second solution can be bracketed by the interval $[-1, -0.1]$ as $f(-1) = 0.3194$ and $f(-0.1) = -2.8238$. The third solution can be bracketed by the interval $[4, 5]$ as $f(4) = 0.6112$ and $f(5) = -0.4974$. $f(x)$ is continuous in each of the bracketing intervals.

(iv) Rearranging $4e^{-x^2/5} = cos(5x) + 2$ gives $f(x) = 4e^{-x^2/5} - cos(5x) - 2$.

```
f = @(x) 4*exp(-x.^2/5) - cos(5*x) - 2;
pltFunc(f, [-2*pi 2*pi], inf);
```





The second plot shows that the equation has 6 solutions. The bracketing intervals are shown in the table below.

$[a, b]$	f(a)	f(b)
$[-2.5, -2]$	-1.8518	0.6364
$[-1.5, -1.25]$	0.2039	-0.0730
$[-1.25, -1]$	-0.0730	0.9913
$[1, 1.25]$	0.9913	-0.0730
$[1.25, 1.5]$	-0.0730	0.2039
$[2, 2.5]$	0.6364	-1.8518

(b) The bisection method used by calling the the `bisectRoot` function.

```
function [sol, i, err] = bisectRoot(f, a, b, tol)
    %bisectRoot Use the bisection method to find roots of the function f
    %   bracketed within the intervals [a, b].
    %
    %Inputs:
    %   f = function handle to function whose root is to be found
    %   a = 1*n array containing all the lower ends of the brackets
    %   where n is the number of roots
    %   b = 1*n array containing all the lower ends of the brackets
    %   where n is the number of roots
    %   tol = absolute error tolerance with which to find the root;
    %   Iteration terminates when the root is known to within +/- tol
    %
    %Usage:
    %   [r, i, err] = bisect(f,a,b,tol) -> returns the approximation to a root
    %   within [a, b], the number of iterations require to find the root
    %   and the final absolute error

    % check if all intervals are correctly defined
    assert(isequal(size(a), size(b)), "");

    % check whether f changes sign
    assert(all(sign(f(a)) ~= sign(f(b))),...
        'f(a) and f(b) should have opposite sign');

    % intialise variables
    % iteration counter
```

```
    i = zeros(size(a));
    % current solution estimate
    sol = (a + b)/2;
    % previous solution estimate
    sol_old = Inf;
    % absolute error
    err = Inf;
    withinTol = zeros(size(a));

    % bisection algorithm:
    % at each iteration, find the half-interval that contains a sign change
    % and relabel the endpoints appropriately
    while any(~withinTol)
        i(~withinTol) = i(~withinTol) + 1;
        sol_old = sol;
        mid = (a + b)/2;

        % mid point is a root
        exactRoot = f(mid) == 0;
        sol(exactRoot) = mid(exactRoot);
        err(exactRoot) = 0;
        withinTol(exactRoot) = true;

        % solution is in first half of interval and mid point not a root
        firstHalf = (sign(f(a)) ~= sign(f(mid))) & ~exactRoot;
        b(firstHalf) = mid(firstHalf);

        % solution is in second half of interval and mid point not a root
        secondHalf = (sign(f(a)) == sign(f(mid))) & ~exactRoot;
        a(secondHalf) = mid(secondHalf);

        % update solutions and errors values that aren't within tolerance
        sol(~withinTol) = (a(~withinTol) + b(~withinTol))/2;
        err(~withinTol) = abs(sol(~withinTol) - sol_old(~withinTol));
        withinTol(err < tol) = true;
    end
end
```

(i) Solutions to $f(x) = x^4 - e^{-x}\cos(x) = 0 \quad x \in [-2\pi, 2\pi]$.

```
f = @(x) x.^4 - exp(-x).*cos(x);
a = [-1.5 0.5];
b = [-1 1];
[r, i, err] = bisectRoot(f, a, b, [5e-8 5e-9])
```

Note the two different tolerances since one root is an order of magnitude larger so requires one less decimal place of accuracy to be accurate to 8 significant figures.

$[a, b]$	Root	# Iterations
$[-1.5, -1]$	-1.0843596	23
$[0.5, 1]$	0.76221106	26

(ii) Solutions to $f(x) = \frac{x^3}{\sin(x)} - 1 = 0 \quad x \in [-2\pi, 2\pi]$.

```
f = @(x) (x.^3)./sin(x) - 1;
a = [-1 0.5];
b = [-0.5 1];
[r, i, err] = bisectRoot(f, a, b, 5e-9)
```

$[a, b]$	Root	# Iterations
$[-1, -0.5]$	-0.92862631	26
$[0.5, 1]$	0.92862631	26

(iii)