```
%% Project 2 - finding the root of a scalar equation
```

```
============================================================
Project 2 - finding the root of a scalar equation
```

```
%% Maddox Gonzalez
```

```
Maddox Gonzalez
```

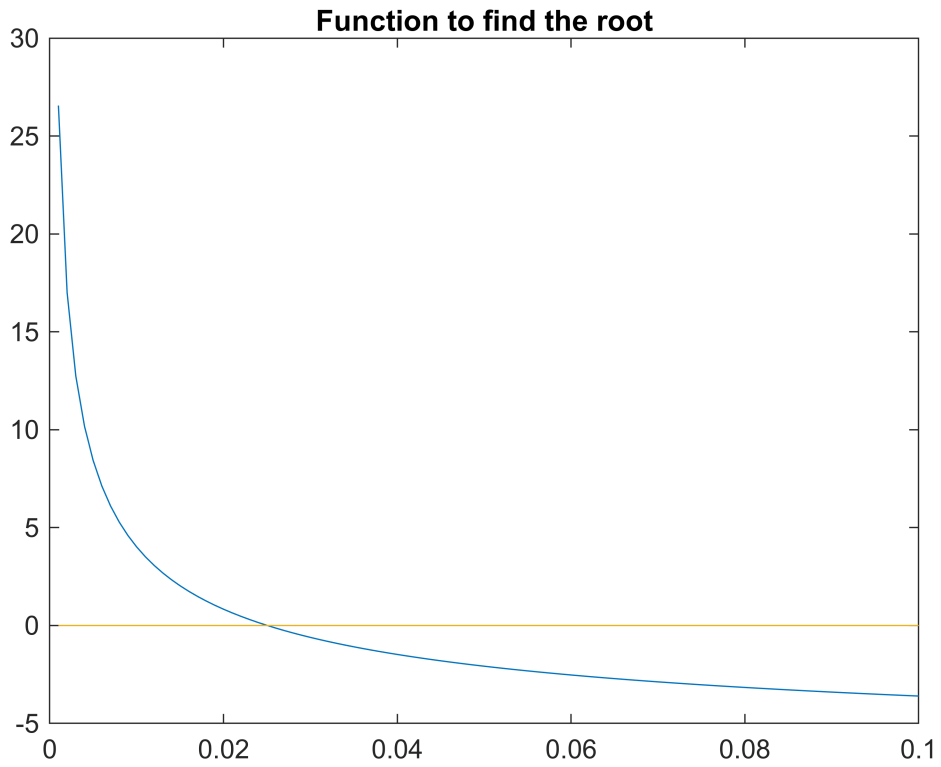
```
%% 2/2/25
```

```
01-Feb-2025
```

```

```

```
============================================================
```

```
clc, clear, close all;

% 6 parameters
L = 3000;
D = 2;
r = 5e-4;
V = 0.45;
Re = 2.8409e4;
g = 32.2;

% Define function, derivative, and head loss function
f = @(x) 1./sqrt(x) + 2*log10((r/3.7) + (2.51./(Re*sqrt(x))));
df = @(x) (-0.5 * x.^-1.5) + (-2 ./ (log(10) * ((r/3.7) + (2.51 ./ (Re *
sqrt(x)))))) .* ((2.51 ./ (2 * Re * x.^1.5)));

% Head loss function
h = @(f) (f * L * V^2) / (D * 2 * g);


max_iter = 1000;
guess1 = 0.001;
guess2 = 0.01;
res_tol = 1e-5;
conv_tol = 1e-5;

% Plot the function
x = 0.001:0.001:0.1;
fval = f(x);
figure
plot(x, fval, x, zeros(length(x)))
title('Function to find the root')
```




```
% Run the solvers for both guess1 and guess2 for NR
fprintf('Newton Raphson (Initial Guess 1)\n')
```

```
Newton Raphson (Initial Guess 1)
```

```
[roots_nr_1, conv_nr_1, res_nr_1, i_nr_1] = ...
    newton_raphson_func(f, df, guess1, conv_tol, res_tol, max_iter);
```

```
iteration x residual convergence
========= ===== ========= ==========
  0 1.000000000e-03 2.6556e+01 0.0000e+00
 1 2.636682969e-03 1.4012e+01 1.6367e-03
 2 6.280122493e-03 6.8126e+00 3.6434e-03
 3 1.266890218e-02 2.8121e+00 6.3888e-03
 4 2.007143516e-02 8.1869e-01 7.4025e-03
 5 2.429982428e-02 1.0760e-01 4.2284e-03
 6 2.503463925e-02 2.3180e-03 7.3481e-04
 7 2.505117306e-02 1.1169e-06 1.6534e-05
 8 2.505118104e-02 2.5935e-13 7.9740e-09
```

```
fprintf('Newton Raphson (Initial Guess 2)\n')
```

```
Newton Raphson (Initial Guess 2)
```

```
[roots_nr_2, conv_nr_2, res_nr_2, i_nr_2] = ...
    newton_raphson_func(f, df, guess2, conv_tol, res_tol, max_iter);
```

```
iteration x residual convergence
========= ===== ========= ==========
  0 1.000000000e-02 4.0161e+00 0.0000e+00
 1 1.746939753e-02 1.3760e+00 7.4694e-03
 2 2.326969290e-02 2.6328e-01 5.8003e-03
 3 2.495745553e-02 1.3163e-02 1.6878e-03
 4 2.505092490e-02 3.5875e-05 9.3469e-05
 5 2.505118104e-02 2.6775e-10 2.5614e-07
```

```
fprintf('\n\n=====================================================\n')
```

```
=====================================================
```

```
fprintf('Secant\n')
```

```
Secant
```

```
[roots_s, conv_s, res_s, i_s] = ...
    secant_func(f, guess1, guess2, conv_tol, res_tol, max_iter);
```

```
iteration x residual convergence
========= ===== ========= ==========
  0 1.000000000e-03 2.6556e+01 0.0000e+00
  1 1.000000000e-02 4.0161e+00 9.0000e-03
 2 1.160355895e-02 3.2437e+00 1.6036e-03
 3 1.833764714e-02 1.1772e+00 6.7341e-03
 4 2.217392673e-02 4.4032e-01 3.8363e-03
 5 2.446626522e-02 8.3348e-02 2.2923e-03
 6 2.500148632e-02 6.9704e-03 5.3522e-04
 7 2.505033207e-02 1.1891e-04 4.8846e-05
 8 2.505117981e-02 1.7221e-07 8.4774e-07
```

```
fprintf('\n=========================================================\n')
```

```
=========================================================
```

```
fprintf('Answers:\n')
```

```
Answers:
```

```
fprintf(' newton (guess 1) = %.9e in %i iterations.\n', roots_nr_1(end), i_nr_1);
```

```
 newton (guess 1) = 2.505118104e-02 in 8 iterations.
```

```
fprintf(' newton (guess 2) = %.9e in %i iterations.\n', roots_nr_2(end), i_nr_2);
```

```
 newton (guess 2) = 2.505118104e-02 in 5 iterations.
```

```
fprintf(' secant = %.9e in %i iterations.\n', roots_s(end), i_s);
```

```
 secant = 2.505117981e-02 in 7 iterations.
```

```
% Head loss calculation for both guesses of NR
head_loss_nr_1 = h(roots_nr_1(end));
fprintf('Newton-Raphson root, guess 1) = %.9f\n',roots_nr_1(end));
```

```
Newton-Raphson root, guess 1) = 0.025051181
```

```
head_loss_nr_2 = h(roots_nr_2(end));
head_loss_s = h(roots_s(end));

fprintf('\nHead Loss Results:\n')
```

```
Head Loss Results:
```

```
fprintf('Head loss (Newton-Raphson, guess 1) = %.9f\n', head_loss_nr_1);
```

```
Head loss (Newton-Raphson, guess 1) = 0.118156774
```

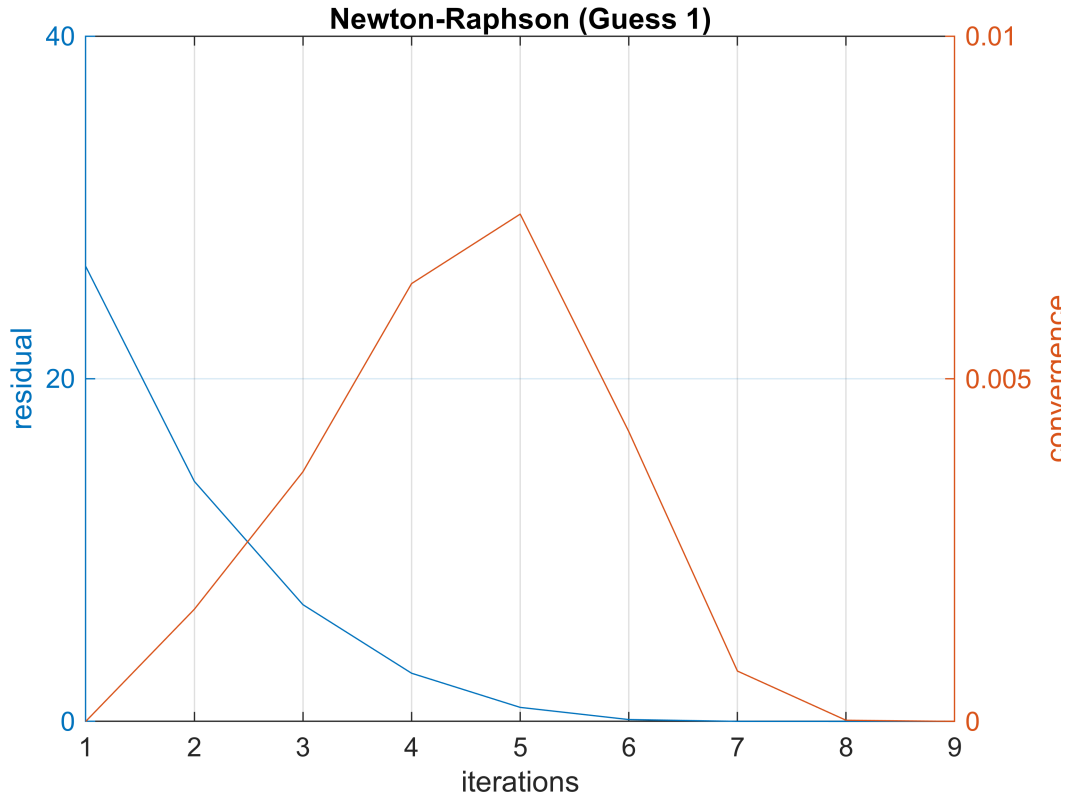
```
fprintf('Head loss (Newton-Raphson, guess 2) = %.9f\n', head_loss_nr_2);
```

```
Head loss (Newton-Raphson, guess 2) = 0.118156774
```

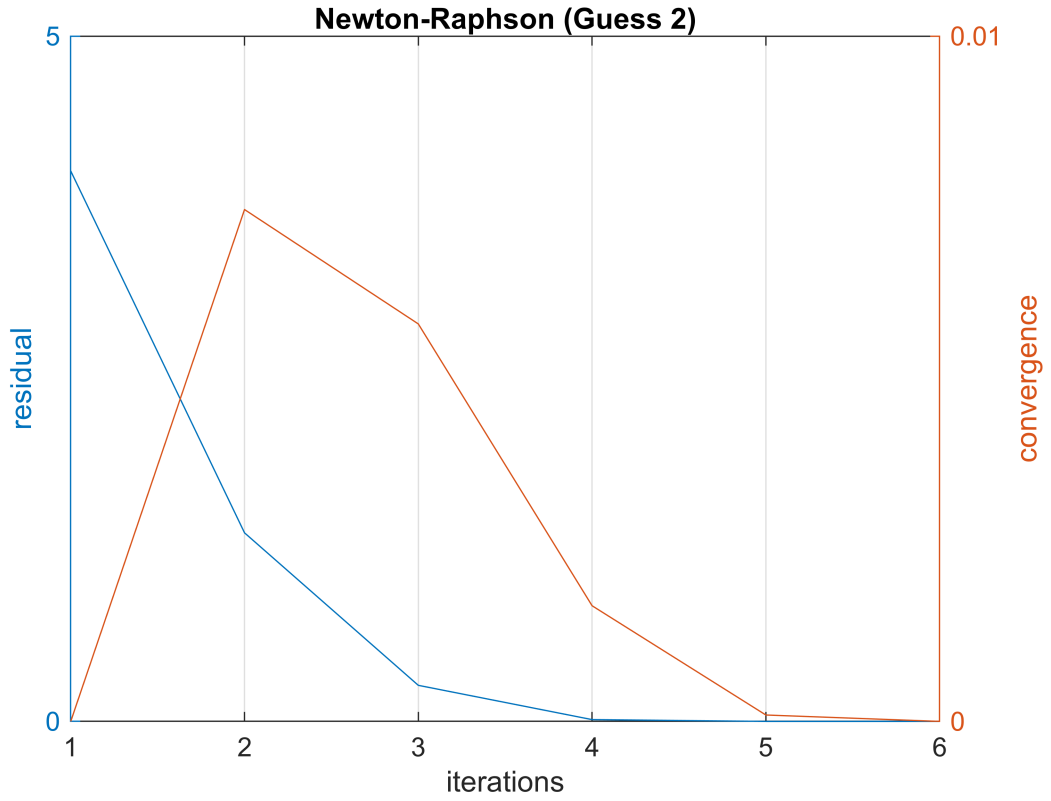
```
fprintf('Head loss (Secant) = %.9f\n', head_loss_s);
```

```
Head loss (Secant) = 0.118156768
```

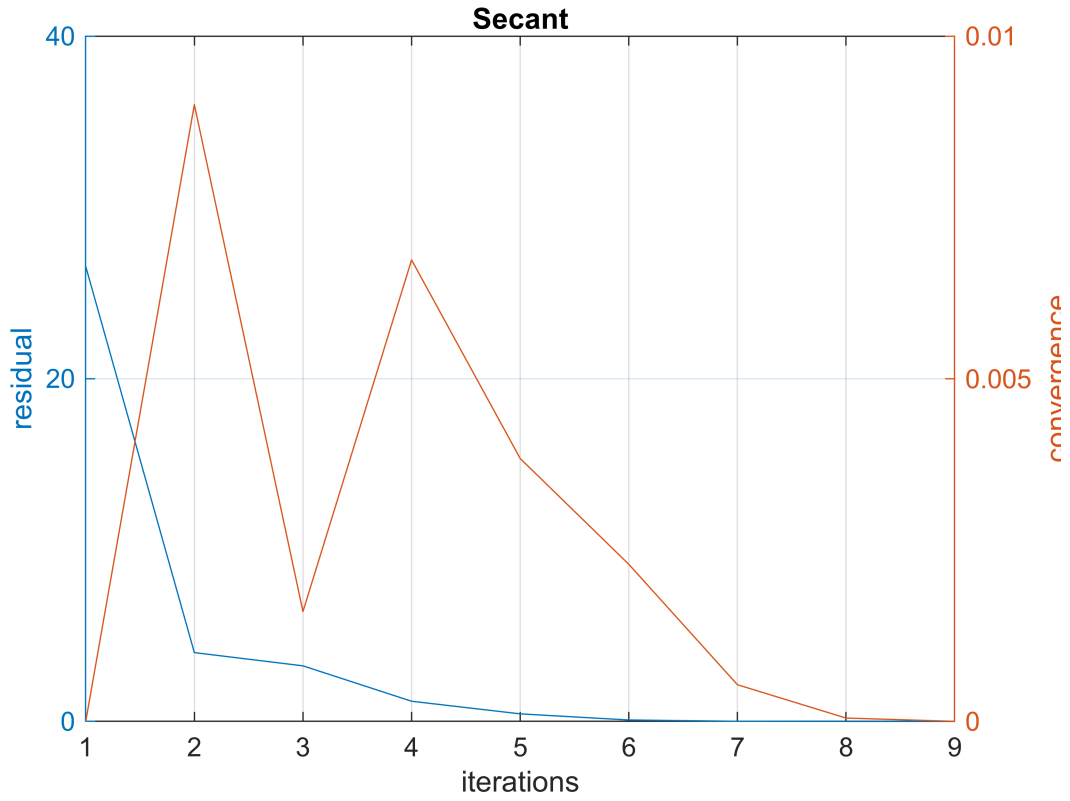
```
% Plot the residual and convergences
plot_iters_nr_1 = 1:length(res_nr_1);
figure
ax = plotyy(plot_iters_nr_1, res_nr_1, plot_iters_nr_1, conv_nr_1);
title('Newton-Raphson (Guess 1)')
xlabel('iterations')
ylabel(ax(1), 'residual')
ylabel(ax(2), 'convergence')
grid on
```




```
plot_iters_nr_2 = 1:length(res_nr_2);
figure
ax = plotyy(plot_iters_nr_2, res_nr_2, plot_iters_nr_2, conv_nr_2);
title('Newton-Raphson (Guess 2)')
xlabel('iterations')
ylabel(ax(1), 'residual')
ylabel(ax(2), 'convergence')
grid on
```




```
plot_iters_s = 1:length(res_s);
figure
ax = plotyy(plot_iters_s, res_s, plot_iters_s, conv_s);
title('Secant')
xlabel('iterations')
ylabel(ax(1), 'residual')
ylabel(ax(2), 'convergence')
grid on
```




```
% Newton-Raphson Function
function [x, conv, res, i] = ...
    newton_raphson_func(f, df, initial_guess, conv_tol, res_tol, max_iter)
    % initial guess
    x(1) = initial_guess;
    % initial convergence
    conv(1) = 0;
    % initial residual
    res(1) = abs(f(x(1)));

    fprintf('iteration x residual convergence\n');
    fprintf('========= ===== ========= ==========\n');
    fprintf(' %2i %.9e %.4e %.4e\n', 0, x(1), res(1), conv(1));

    % initialize the iteration counter
    i = 1;

    % iterate using our formula
    while i < max_iter
        fval = f(x(i));
        dfval = df(x(i));

        if (dfval == 0) || (dfval == NaN)
            error("!!! FAILED. The derivative value is %i. Try another guess.",
dfval)
```

```
            break;
        end

        x(i + 1) = x(i) - (fval / dfval);  % apply Newton-Raphson formula
        conv(i + 1) = abs(x(i + 1) - x(i));  % calculate iterative convergence
        res(i + 1) = abs(f(x(i + 1)));  % calculate the residual

        fprintf(' %i %.9e %.4e %.4e\n', i, x(i + 1), res(i + 1), conv(i + 1));

        % check for convergence
        if conv(i + 1) < conv_tol && res(i + 1) < res_tol
            break;
        end

        i = i + 1;  % increment our iteration counter
    end
end

% Secant Function
function [x, conv, res, i] = ...
    secant_func(f, initial_guess1, initial_guess2, conv_tol, res_tol, max_iter)
    % initial guess
    x(1) = initial_guess1;
    x(2) = initial_guess2;

    % initial convergence
    conv(1) = 0;
    conv(2) = abs(x(2) - x(1));

    % initial residual
    res(1) = abs(f(x(1)));
    res(2) = abs(f(x(2)));

    fprintf('iteration x residual convergence\n');
    fprintf('========= ===== ========= ==========\n');
    fprintf(' %2i %.9e %.4e %.4e\n', 0, x(1), res(1), conv(1));
    fprintf(' %2i %.9e %.4e %.4e\n', 1, x(2), res(2), conv(2));

    % initialize the iteration counter
    i = 2;

    % iterate using our formula
    while i < max_iter
        fval = f(x(i));
        dfval = (f(x(i)) - f(x(i-1))) / (x(i) - x(i-1));  % secant approximation
for derivative

        if (dfval == 0) || (dfval == NaN)
            error("!!! FAILED. The derivative value is %i. Try another guess.",
dfval)
```

```
            break;
        end

        x(i + 1) = x(i) - (fval / dfval);  % apply Secant formula
        conv(i + 1) = abs(x(i + 1) - x(i));  % calculate iterative convergence
        res(i + 1) = abs(f(x(i + 1)));  % calculate the residual

        fprintf(' %i %.9e %.4e %.4e\n', i, x(i + 1), res(i + 1), conv(i + 1));

        % check for convergence
        if conv(i + 1) < conv_tol && res(i + 1) < res_tol
            break;
        end

        i = i + 1;  % increment our iteration counter
    end
    i = i - 1;  % final iteration count
end
```