%% Class 3 classdef set\_3 methods(Static)

%% Problem 1: Secant Method

%

% \* input command: set\_3.secant\_method(13, 16, .001)

% \* output: 14.2325

%

function [sol] = secant\_method(x\_p, x\_n, err\_acc)

error = abs(x\_n- x\_p) / x\_n;

function[y] = fxn(x)

y = atan(exp(sin(1 / (x + 20)))) - .8;

end

while (error > err\_acc) x = [x\_p x\_n];

f = [fxn(x\_n) -fxn(x\_p)]'; sol = (x\*f) / (f(1) + f(2)); error = abs(x\_n- x\_p) / x\_n;

x\_p = x\_n;

x\_n = sol;

end

end

%% Problem 2: Fixed Point

%

% \* input command: set\_3.fixed\_point(3, .001)

% \* output: 2.5222

%

function [x\_root] = fixed\_point(x\_0, err\_accept)

error = 100;

% If I understand the function's alg correctly. We must solve

% for x using the highest degree polynomial of x since it will

% create a decreasing funciton as x, in a particular set of

% directions, increases so solving for x = .5( x^3 - 11) and

% x = (2\*x + 11) / (x^2) could cause issues function y=gxn(x)

y = (2\*x + 11)^ (1/3);

end

while (error > err\_accept)

x\_root = gxn(x\_0);

error = abs(x\_root - x\_0) / x\_root;

x\_0 = x\_root;

end

x\_root = x\_0; end

%% Problem 3: Second Dimension Error % \* input command:

% >> b2 = .9:.01:1.1; % >> set\_3.solution\_plot(b2) % \* output:

%

%

% <<../../sub/p3.png>>

%

%

function solution\_plot(b2)

x1 = -841 + 842 \* b2; x2 = 921 - 922 \* b2;

x3 = 3 - 3 \* b2;

f1 = figure(1); hold on

plot(b2, x1, '-r');

plot(b2, x2, '-g');

plot(b2, x3, '-b');

legend("x1", "x2", "x3");

hold off

%uiwait(f1);

end end

end

A graph with lines and numbers

Description automatically generated