Numerical Analysis (089180) A.Y. 2022-2023
prof. Simona Perotto Luca Liverotti
simona.perotto@polimi.it luca.liverotti@polimi.it

Lab 2 – Homework Solutions

September 23, 2022

A Homework

Homework 2.1 (Bisection method). Consider the following equation

$$\cot(x) = \frac{x^2 - 1}{2x}.$$

a. Define three intervals containing the three smallest positive roots.

b. Find the biggest root among the ones at item a), using bisection method with tolerance equal to 10^{-6}.

c. Plot the evolution of the error as a function of the number of the iterations.

Solution Homework 2.1.

```
function rootfinding_function_plot(f, a, b, new_figure)

  % Check if a new graphic window is requested
  if ((nargin < 4) || new_figure)
    figure
  end

  hold on, box on
  x_plot = linspace(a, b, 1000);

  % Plot the function
  plot(x_plot, f(x_plot), 'LineWidth', 2)

  % Plot a zero-line
  plot(x_plot, 0*x_plot, 'k-', 'LineWidth', 1)

  xlabel('x','FontSize', 16)
  ylabel('f(x)','FontSize', 16)
  set(gca,'FontSize', 16)
  set(gca,'LineWidth', 1.5)

end

% HOMEWORK 2.1

f = @(x) cot(x);
g = @(x) (x.^2 - 1)./(2*x);

epsilon = 0.01;
rootfinding_function_plot(f, 0+epsilon, pi-epsilon, true);
rootfinding_function_plot(f, pi+epsilon, 2*pi-epsilon, false);
rootfinding_function_plot(f, 2*pi+epsilon, 3*pi-epsilon, false);
rootfinding_function_plot(g, 0, 3*pi, false);

% Three intervals for the roots are then [1,2], [3,4] and [6,7].
```

```
tol = 1e-6;
h = @(x) cot(x) - (x.^2 - 1)./(2*x);

rootfinding_function_plot(h, 0+epsilon, pi-epsilon, true);
rootfinding_function_plot(h, pi+epsilon, 2*pi-epsilon, false);
rootfinding_function_plot(h, 2*pi+epsilon, 3*pi-epsilon, false);

I = [6.5 7] % cannot use [6, 7] since h is discontinuous there
[xi, x] = bisection(h, I(1), I(2), tol);
xi
iter = numel(x)

% Since the exact value of the roots is NOT known, we perform
% again the bisection method with a very small tolerance, and consider
% the final approximation as the exact root of the equation.

figure

% First root
[xiex, xex] = bisection(h, I(1), I(2), 1e-10*tol);
xiex
err = abs(x - xiex);

subplot(1,2,1)
semilogy(err, 'LineWidth',2)
box on, hold on
semilogy(tol*ones(iter,1), 'r--', 'LineWidth',2)
xlim([0 iter+1])
set(gca,'FontSize',16)
xlabel('Iteration','FontSize',16)
ylabel('Error','FontSize',16)
```

Homework 2.2 (Bisection method). Consider the following equation

$$e^{-(x-2)^2} = 1 - e^{x-4}$$

a. Plot fuction $f(x) = 0$ obtained from the above equation by writing all the therms in the first member and determine the number of real solutions.

b. Apply the bisection method by choosing a tolerance equal to 10^{-3} and initial interval $I_1 = [0,5], I_2 = [1,6], I_3 = [1,5]$ and $I_4 = [-1,5]$. Can you find all the roots?

Solution Homework 2.2.

```
% HOMEWORK 2.2

f = @(x) exp(-(x-2).^2) + exp(x-4) - 1;
rootfinding_function_plot(f, 0, 5);

% There are three solutions in the interval [1, 5].

tol = 1e-6;

I1 = [0 5];
[xi1, x_iter1] = bisection(f, I1(1), I1(2), tol);
xi1

I2 = [1 6];
[xi2, x_iter2] = bisection(f, I2(1), I2(2), tol);
xi2

I3 = [1 5];
[xi3, x_iter3] = bisection(f, I3(1), I3(2), tol);
xi3

I4 = [-1 5];
```

```
[xi4, x_iter4] = bisection(f, I4(1), I4(2), tol);
xi4


% In the first and fourth interval the bisection method converges to the root with
% minimum magnitude; similarly, in the second and third interval the bisection method
% converges to the root with maximum magnitude. The second root cannot be reached.
%
% This is caused by the iterative procedure that halves the interval: at the
% very first iteration the half-interval that contains the second root is discarded.

bisection_plot(f, I2(1), I2(2), tol);
bisection_plot(f, I3(1), I3(2), tol);

% This example shows that the choice of the initial interval is important.
```

Homework 2.3 (Newton method). a. Approximate the root x^* of $\tan(x) = 2x$ in the interval $\left(0, \frac{\pi}{2}\right)$ up to a tolerance equal to 10^{-15}.

b. Approximate the zero of $\sin(x)$ in the interval $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$ using Newton method and starting from the value $x^{(0)} = x^*$. What do you expect? What do you observe?

Solution Homework 2.3.

```
function [xi, x_iter] = newton25(f, df, x0, tol, maxit, multiplicity)
 %NEWTON Find a root of the equation f(x) = 0 using the Newton method, starting from the
 initial guess x0.
 %
 %  [xi, x_iter] = NEWTON(f, df, x0, tol, maxit, multiplicity)
 %
 %  Inputs : f  = function handle to the function f(x)
 %         df   = function handle to the derivative of the function f(x)
 %         x0   = initial guess
 %         tol  = requested tolerance
 %         maxit = maximum number of iterations
 %         multiplicity = multiplicity of the root
 %  Output :
 %      xi = approximation of the root
 %      x_iter  = vector of the approximations of the root at each step

 if (nargin < 6)
  multiplicity = 1;
 end

 x_iter(1) = x0;

 for (iter = 1:maxit)
  newton_method = @(x) x - multiplicity*f(x)/df(x);
  x_iter(iter+1) = newton_method(x_iter(iter));

  if (abs (x_iter(iter+1) - x_iter(iter)) < tol)
   break;
  end

 end

 xi = x_iter(end);

end

% HOMEWORK 2.3

f = @(x) tan(x) - 2*x;
epsilon = 0.01;
rootfinding_function_plot(f, 0+epsilon, pi/2-epsilon, true);

% From the plot we choose the interval I = [1, 1.5] that ensures the validity of the Bolzano's
% theorem, and we can use e.g. the bisection method to find x*.
```

```
xstar = bisection(f, 1, 1.5, 1e-15);
```

```
% The Newton method for the approximation of sin(x) = 0 is
%
% x{(k+1)} = x{(k)} - {sin(x{(k)})}/{cos(x{(k)})} = x{(k)} - tan(x{(k)}).
%
% Therefore, if x{(0)} = x*
%
% x{(1)} = x{(0)} - tan(x{(0)}) = x* - tan(x*) = x* - 2x* = -x* = -x{(0)}\\
% x{(2)} = x{(1)} - tan(x{(1)}) = -x* + tan(x*) = -x* + 2x* = x* = x{(0)}
%
% The method is describing the orbit {x*, - x*}, and thus cannot converge!
```

```
g = @(x) sin(x);
dg = @(x) cos(x);

[xi, x_iter] = newton25(g, dg, xstar, 1e-6, 25);
x_iter

% The first 10 iterations of the Newton method seem to describe the orbit {x*, - x*}, but
% after the first few iterations some numerical error arises and this makes the algorithm move
 from the orbit and then to converge to the solution.
```

Homework 2.4 (Newton method). Consider the following function

$$f(x) = e^{ax} - 1 = 0, \qquad a \neq 0.$$

For any value of the parameter a the unique solution is clearly $\xi = 0$.

a. Consider the case of $a = 200$ and apply Newton method starting from $x^{(0)} = 1$ with tolerance equal to 10^{-3} and 10^{-12}.

b. Repeat item a) for $a = 10^{-3}$.

c. For both items b) and c), compare the absolute value of the residual with the error. Justify the obtained results.

Solution Homework 2.4.

```
% HOMEWORK 2.4

a = 200;
f = @(x) exp(a*x) - 1;
df = @(x) a*exp(a*x);

maxit = 1000;
x0 = 1;

[xi, x] = newton25(f, df, x0, 1e-3, maxit);
err_1_3 = abs(xi)
absres_1_3 = abs( f(xi) )

[xi, x] = newton25(f, df, x0, 1e-12, maxit);
err_1_12 = abs(xi)
absres_1_12 = abs( f(xi) )

%%
a = 1e-3;
f = @(x) exp(a*x) - 1;
df = @(x) a*exp(a*x);

maxit = 1000;
x0 = 1;

[xi, x] = newton25(f, df, x0, 1e-3, maxit);
```

```
err_2_3 = abs(xi)
absres_2_3 = abs( f(xi) )

[xi, x] = newton25(f, df, x0, 1e-12, maxit);
err_2_12 = abs(xi)
absres_2_12 = abs( f(xi) )


disp('   tol      absres    error')
disp([1e-3 absres_1_3 err_1_3;
    1e-12 absres_1_12 err_1_12])

disp('   tol      absres    error')
disp([1e-3 absres_2_3 err_2_3;
    1e-12 absres_2_12 err_2_12])


% We know that abs{f(x)} < abs{f'(xi)}abs{x - xi}
% Hence if the derivative of f at xi is big the residual can be big
% while the solution is already well approximated.
% Conversely if the derivative is small, the residual decreases fast and
% the convergence criterion can result to be too weak.
```

Homework 2.5 (Newton method). The equation $x^3 - x^2 - x + 1 = 0$ has a double root at $\xi = 1$. Given $x^{(0)} = 2$, does Newton method converge to ξ? How can you improve the method for approximating ξ?

Solution Homework 2.5.

```
clear all
close all
clc
% HOMEWORK 2.5

f = @(x) x.^3 - x.^2 - x + 1;
df = @(x) 3*x.^2 - 2*x - 1;
rootfinding_function_plot(f, -2, 2, true);

tol = 1e-6
nmax = 100;
[xin, xn] = newton25(f, df, 2, tol, nmax);
xin
itern = numel(xn)

[xinmod, xnmod] = newton25(f, df, 2, tol, nmax, 2);
xinmod
iternmod = numel(xnmod)

err_newton = abs(xn - 1);
err_newton2 = abs(xnmod - 1);

figure
semilogy(err_newton, 'bs--','LineWidth',2)
hold on, box on
semilogy(err_newton2, 'ro-','LineWidth',2)
axis([0 itern+1 1e-12 10])
set(gca,'FontSize',16)
set(gca,'LineWidth',1.5)
xlabel('iterations','FontSize',16)
ylabel('error','FontSize',16)
h = legend('Newton', 'Modified Newton');
set(h,'FontSize',16)
```