ME 594 – Numerical Methods – HW04

Viral Panchal | Due Date: 10/19

‘I pledge my honor that I have abided by the stevens honor system’

Table

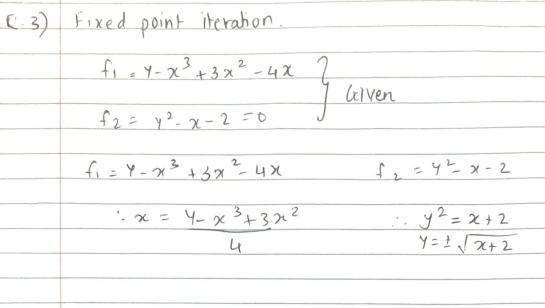
Description automatically generated with medium confidence

A piece of paper with writing

Description automatically generated with low confidence

Text, letter

Description automatically generated

’

# Q3. Program

* ***Script for fixed point algorithm***

% Function for fixed point algorithm

function [P,rel\_error,n\_iters] = fixed\_point(G,P,tol,max\_iter)

% rel\_error: relative error in the solution

% P: Initial guess during input and Fixed point approximation during

% output.

% G = non linear system saved in it's own function file.

n = length(P);

for i = 1:max\_iter

x = feval(G,P);

error = norm(x-P);

rel\_error = error/(norm(x) + eps);

P = x;

n\_iters = i;

if (rel\_error<tol)

break

end

end

* ***Script for non-linear system G1***

function z = G1(x)

z = zeros(1,2);

z(1) = (x(2)-(x(1)^3)+(3\*x(1)^2))/4;

z(2) = -sqrt(x(1)+2);

end

* ***Script for non-linear system G2***

function z = G2(x)

z = zeros(1,2);

z(1) = (x(2)-(x(1)^3)+(3\*x(1)^2))/4;

z(2) = sqrt(x(1)+2);

end

* ***Driver to run Q3***

% Q3 driver  
  
close all  
clear all  
clc  
  
tol = 10^(-6);  
max\_iter = 100;  
P1 = [-0.3 -1.3];  
[P1,rel\_error\_1,n\_iters\_1] = fixed\_point('G1',P1,tol,max\_iter);  
fprintf("First point:\n");  
disp(P1)  
fprintf("Relative error: \n");  
disp(rel\_error\_1)  
fprintf("Number of iterations: \n");  
disp(n\_iters\_1)  
  
P2 = [0.3 1];  
[P2,rel\_error\_2,n\_iters\_2] = fixed\_point('G2',P2,tol,max\_iter);  
fprintf("Second point:\n");  
disp(P2)  
fprintf("Relative error: \n");  
disp(rel\_error\_2)  
fprintf("Number of iterations: \n");  
disp(n\_iters\_2)

* ***MATLAB output:***

First point:  
 -0.2695 -1.3155  
  
Relative error:   
 6.4627e-07  
  
Number of iterations:   
 11  
  
Second point:  
 0.6699 1.6340  
  
Relative error:   
 8.4306e-07  
  
Number of iterations:   
 42

[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

# Q4. Newton’s method to solve given equations:

* ***MATLAB Program:***
* ***Script defining given equations:***

% Given equations for the system

function z = f(x)

a = x(1);

b = x(2);

c = x(3);

z = zeros(3,1);

z(1) = a^2 - a + b^2 + c^2 -5;

z(2) = a^2 + b^2 - b + c^2 -4;

z(3) = a^2 + b^2 + c^2 + c - 6;

end

* ***Script for getting Jacobian:***

% Function to jacobian

function w = Jacobian(x)

a = x(1);

b = x(2);

c = x(3);

w = [2\*a-1 2\*b 2\*c; 2\*a 2\*b-1 2\*c; 2\*a 2\*b 2\*c+1];

end

* ***Script for Newton’s method:***

% Function for newton's method

function [P,iter,rel\_error,est\_error] = Newton(f,Jacobian,P,eps,max\_iter)

% P: initial guess during input and approximation to solution during output

% f: calling given equations

% Jacobian: calling output of jacobian.m

n=length(P);

y=feval(f,P);

for i = 1:max\_iter

jacobian = feval(Jacobian,P);

aug\_mat = [jacobian -y];

delta\_P = zeros(n,1);

for j = 1:n-1

[max\_element,k] = max(abs(aug\_mat(j:n,j)));

if (k>1)

temp\_row = aug\_mat(j,:);

aug\_mat(j,:) = aug\_mat(j+k-1,:);

aug\_mat(j+k-1,:) = temp\_row;

end

for l = j+1:n

pivot = aug\_mat(l,j)/aug\_mat(j,j);

aug\_mat(l,:) = aug\_mat(l,:)-pivot.\*aug\_mat(j,:);

end

end

% Not to do back substitution

delta\_P(n) = aug\_mat(n,n+1)/aug\_mat(n,n);

for m = n-1:-1:1

delta\_P(m) = (aug\_mat(m,n+1)-aug\_mat(m,m+1:n)\*delta\_P(m+1:n))/aug\_mat(m,m);

end

temp\_point = P + delta\_P;

y=feval(f,temp\_point);

error = norm(temp\_point-P);

rel\_error = error/(norm(temp\_point)+eps);

est\_error = norm(y);

P = temp\_point;

iter = i;

if(rel\_error<eps || (est\_error<eps))

break

end

end

* ***Driver for above function:***

%Q4 driver  
  
close all  
clear all  
clc  
  
eps = 10^-8;  
max\_iter = 100;  
fprintf('Initial estimated point:\n');  
P = [0 0 0]';  
disp(P)  
  
fprintf('Numerical solution: \n');  
[P,iter,rel\_error,est\_error] = Newton('f','Jacobian', P, eps, max\_iter)

* ***MATLAB output:***

Initial estimated point:  
 0  
 0  
 0  
  
Numerical solution:   
  
P =  
  
 -0.8471  
 0.1529  
 1.8471  
  
  
iter =  
  
 7  
  
  
rel\_error =  
  
 5.1387e-06  
  
  
est\_error =  
  
 1.8994e-10

* [*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)