

**Addis Ababa University**

**Department of Mathematics**

**Course Title: Applied Numerical Analysis (Math 3221)**

**Assignment**

**Group members:**

**1. Yohannes Wale – UGR/1232/14**

2. Kaleab Ayenew – UGR/2302/14

3. Ebsa Tsegaye - UGR/6942/13

**Submitted to:**

1. **Newton-Raphson Method for Solving Systems of a single Nonlinear Equation**

function main

func = @(x) sin(x) - 0.5; % f(x) = sin(x) - 0.5

dfunc = @(x) cos(x); % f'(x) = cos(x)

x0 = 1; % Initial guess

tol = 1e-6; % Tolerance

max\_iter = 100; % Maximum iterations

root = newtonRaphson(func, dfunc, x0, tol, max\_iter);

disp(['Root found: ', num2str(root)]);

end

function root = newtonRaphson(func, dfunc, x0, tol, max\_iter)

x = x0; iter = 0;

fprintf('Iteration\tValue of x\n');

fprintf('-----------------------------\n');

fprintf('%d\t\t%.6f\n', iter, x);

while iter < max\_iter

fx = func(x); dfx = dfunc(x);

if dfx == 0, error('Derivative is zero. No solution found.'); end

**x\_new = x - fx / dfx;**

iter = iter + 1;

fprintf('%d\t\t%.6f\n', iter, x\_new);

if abs(x\_new - x) < tol

fprintf('Converged to root: %.6f after %d iterations\n', x\_new, iter);

root = x\_new; return;

end

x = x\_new;

end

warning('Maximum iterations reached without convergence.');

root = x;

end

1. **Gauss-Seidel Method for Solving a System of Linear Equations**

function main

n = 3; % Number of variables

A = [4 1 0; 1 2 2; 9 0 5]; % Matrix A

b = [8; 1; 1]; % Vector b

x0 = [0; 0; 0]; % Initial guess

tol = 1e-6; % Tolerance

maxIter = 100; % Maximum iterations

x = gaussSeidel(A, b, x0, tol, maxIter); % Call Gauss-Seidel method

fprintf('The solution vector x is:\n');

disp(x);

end

function x = gaussSeidel(A, b, x0, tol, maxIter)

n = length(b); x = x0; iter = 0;

while iter < maxIter

x\_old = x;

for i = 1:n

sum1 = 0; sum2 = 0;

for j = 1:i-1, sum1 = sum1 + A(i, j) \* x(j); end

for j = i+1:n, sum2 = sum2 + A(i, j) \* x\_old(j); end

**x(i) = (b(i) - sum1 - sum2) / A(i, i);**

end

if norm(x - x\_old, inf) < tol, break; end

iter = iter + 1;

end

if iter == maxIter

fprintf('Maximum iterations reached. Solution may not be accurate.\n');

else

fprintf('Solution found in %d iterations.\n', iter);

end

end

% Run the main function

Main

### Newton-Raphson Method for Solving a System of Nonlinear Equations in MATLAB

function main

F = @(X) [X(1)^2 - X(2)^2 - 4; X(1)^2 + X(2)^2 - 16]; % Define the system

X0 = [2.828; 2.828]; % Initial guess

tol = 1e-6; % Tolerance

maxIter = 100; % Maximum iterations

[X, iter] = newtonRaphsonSys(F, X0, tol, maxIter); % Call Newton-Raphson

fprintf('The solution vector X is:\n'); disp(X); fprintf('Number of iterations: %d\n', iter);

end

function J = calculateJacobian(F, X)

n = length(X); J = zeros(n, n); h = 1e-6; % Jacobian calculation

for i = 1:n

Xf = X; Xb = X; Xf(i) = Xf(i) + h; Xb(i) = Xb(i) - h;

J(:, i) = (F(Xf) - F(Xb)) / (2 \* h);

end

end

function [X, iter] = newtonRaphsonSys(F, X0, tol, maxIter)

X = X0; iter = 0;

while iter < maxIter

FX = F(X); JX = calculateJacobian(F, X); % Evaluate function and Jacobian

**DX = JX \ FX; X = X - DX;** % Update solution

if norm(FX, inf) < tol, break; end % Check convergence

iter = iter + 1;

end

if iter == maxIter

fprintf('Maximum iterations reached. Solution may not be accurate.\n');

else

fprintf('Solution found in %d iterations.\n', iter);

end

end

% Run the main function

main