

---

# Table of Contents

Assignemnt .....	1
Part 1 (Preprocessing) .....	1
Part 2 (Processing / Using the function) .....	2
Part 3 (post processing) .....	2

# Assignemnt

Jacobi Itterator

```
% Name      : Mohamed Mafaz
% Roll Number : AM25M009
% Department  : Applied Mechanics
```

```
clc;
clear;
```

```
loops_taken = 0;
```

```
A = [4   1  -1;
      1  -5  -1;
      2  -1  -6];
```

```
B = [13; -8; -2];
```

```
X = [0; 0; 0];
X_new = [0; 0; 0];
```

```
tolerence = 1e-12;
relative_error = 0;
```

## Part 1 (Preprocessing)

Checking if Diagonally Dominant

```
function [] = Diag_dom(A)
    diag_dom = 0;
    for j = 1:length(A)
        sum = 0;
        for i = 1:length(A)
            if i ~= j
                sum = sum + abs(A(j, i)); % abs value of sum of non diagonal
elements
            end
        end
        if sum > abs(A(j, j)) % Comparing to diagonal elements
            diag_dom = diag_dom + 1
        end
    end
end
```

---

```

    if diag_dom > 0
        fprintf('Matrix is not Diagonally dominant :(\n\n')
    else
        fprintf('Matrix is Diagonally dominant :)\n\n')
    end
end

Diag_dom(A)

Matrix is Diagonally dominant :)

```

## Part 2 (Processing / Using the function)

```

while (relative_error > tolerance) || loops_taken == 0

    for j = 1:length(A)
        sum = 0;
        for i = 1:length(A)
            if i ~= j
                sum = sum + (A(j, i) * X(i));           % Sum of non
diagnol elements
            end
        end
        X_new(j) = (B(j) - sum) / A(j, j);             % b - sum
                                                    % Jacobi doesnt

use new values right away in the same loop
    end

    relative_error = max(abs(X_new - X) ./ (X_new + 1e-9)); % Calculating
relative error

    X = X_new;                                         % Copying new
array to actual array of initial guesses
    loops_taken = loops_taken + 1;

```

## Part 3 (post processing)

Printing it out

```

fprintf("Loop: %d |", loops_taken)
for i = 1:length(X)
    fprintf("    X_%d: %f |", i, X(i))
end
fprintf("\n")

```

```

Loop: 1 |    X_1: 3.250000 |    X_2: 1.600000 |    X_3: 0.333333
/

Loop: 2 |    X_1: 2.933333 |    X_2: 2.183333 |    X_3: 1.150000
/

```

---

Loop: 3	/	x_1: 2.991667	/	x_2: 1.956667	/	x_3: 0.947222
/						
Loop: 4	/	x_1: 2.997639	/	x_2: 2.008889	/	x_3: 1.004444
/						
Loop: 5	/	x_1: 2.998889	/	x_2: 1.998639	/	x_3: 0.997731
/						
Loop: 6	/	x_1: 2.999773	/	x_2: 2.000231	/	x_3: 0.999856
/						
Loop: 7	/	x_1: 2.999906	/	x_2: 1.999983	/	x_3: 0.999886
/						
Loop: 8	/	x_1: 2.999976	/	x_2: 2.000004	/	x_3: 0.999972
/						
Loop: 9	/	x_1: 2.999992	/	x_2: 2.000001	/	x_3: 0.999991
/						
Loop: 10	/	x_1: 2.999998	/	x_2: 2.000000	/	x_3: 0.999997
/						
Loop: 11	/	x_1: 2.999999	/	x_2: 2.000000	/	x_3: 0.999999
/						
Loop: 12	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						
Loop: 13	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						
Loop: 14	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						
Loop: 15	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						
Loop: 16	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						
Loop: 17	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						
Loop: 18	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						
Loop: 19	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						
Loop: 20	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						
Loop: 21	/	x_1: 3.000000	/	x_2: 2.000000	/	x_3: 1.000000
/						

---

---

```
Loop: 22  /   x_1: 3.000000  /       x_2: 2.000000  /       x_3: 1.000000
/
Loop: 23  /   x_1: 3.000000  /       x_2: 2.000000  /       x_3: 1.000000
/
Loop: 24  /   x_1: 3.000000  /       x_2: 2.000000  /       x_3: 1.000000
/
end

fprintf("\nLoops Taken: %d\n", loops_taken)
fprintf("\nRelative Error: %d\n", relative_error)
```

*Loops Taken: 24*

*Relative Error: 4.013456e-13*

*Published with MATLAB® R2025a*