CL249 COMPUTATIONAL METHODS LAB

ASSIGNMENT 3 : LU DECOMPOSITION

Submitted by :

AYUSHMAN CHOUDHARY

Q. Write a program based on the LU decomposition method to find the inverse of a given A matrix.

In main code, read A matrix from data file. Call code to find L and U matrix using Gauss Elimination method. Write function which takes L, U, B as input to solve

$$Ax = b$$

and returns x vector.

Method : LU decomposition

Using Gauss-Elimination method, we represent A matrix as

$$A = LU$$

S.t.

L is a lower triangular matrix with diagonal elements = 1.

U is an upper triangular matrix

Now, we know that for any inversible matrix A,

$$A A^{-1} = I$$

Hence, we apply LU decomposition separately for every column of I matric i.e.

$$A x_i = b_i$$

where b_i is i^{th} column of I and x_i is i^{th} column of A^{-1}.

$$\Rightarrow \quad L U x_i = b_i$$

$$U x_i = d_i \quad \text{(back substitution)}$$

$$L d_i = b_i \quad \text{(forward substitution)}$$

PSEUDOCODE

main.m

Take matrix A as input

Calculate size of matrix A (size_A)

Feed A into LU_calc f^n to get the matrix L and U and count operations.

LU_calc.m

Initialize L matrix to I of size_A.

Initialize counter to 0

Loop i: 1 to size_A

Ensure pivoting and max. diagonal element s.t.

$a_{ii} > \forall\ a_{ji}$ where $j = i+1$ to size_A

$a_{ii} \neq 0$

Loop j: i+1 to size_A

$$factor(j,i) = \frac{A(j,i)}{A(i,i)}$$

$$L(j,i) = factor(j,i)$$

$$A(j,k) = A(j,k) - factor(j,i) \cdot A(i,k)$$

(k = 1 to size_A)

Update counter

U = A

main.m

loop $i = 1$ to Size_A

b is i^{th} column of I

feed L, U, B to inverse_calc

to get x(i^{th} col) and counter.

inverse_calc.m

Initialize d column vector to ZERO

of length Size_A.

Initialize counter = 0

loop i : 1 to Size_A

$$d(i) = B(i) - \sum_{j=1}^{i-1} L(i,j) \cdot d(j)$$

update counter.

Initialize x column vector to ZERO

loop i : Size_A to 1 :

$$x(i) = \frac{d(i) - \sum_{j=i+1}^{Size\_A} U(i,j) \cdot x(j)}{U(i,i)}$$

update counter.

main.m

Print x

Print counter

Comments and Remarks.

Gauss elimination method requires 2570 operation per b vector.

Hence, total operations for GEM = 2570 × 15
= 38,550

LU decomposition method requires ~ 8870 operations

Hence, when iterations for b vector is comparable to size of A matrix,

LU decomposition is better than Gauss Elimination.

```
clear all;
clc

% Finding the inverse of matrix A using LU Decomposition

% A= [9 -4 1 0 0 0 0 0 0 0 0 0 0 0 0
%     -4 6 -4 1 0 0 0 0 0 0 0 0 0 0 0
%     1 -4 6 -4 1 0 0 0 0 0 0 0 0 0 0
%     0 1 -4 6 -4 1 0 0 0 0 0 0 0 0 0
%     0 0 1 -4 6 -4 1 0 0 0 0 0 0 0 0
%     0 0 0 1 -4 6 -4 1 0 0 0 0 0 0 0
%     0 0 0 0 1 -4 6 -4 1 0 0 0 0 0 0
%     0 0 0 0 0 1 -4 6 -4 1 0 0 0 0 0
%     0 0 0 0 0 0 1 -4 6 -4 1 0 0 0 0
%     0 0 0 0 0 0 0 1 -4 6 -4 1 0 0 0
%     0 0 0 0 0 0 0 0 1 -4 6 -4 1 0 0
%     0 0 0 0 0 0 0 0 0 1 -4 6 -4 1 0
%     0 0 0 0 0 0 0 0 0 0 1 -4 6 -4 1
%     0 0 0 0 0 0 0 0 0 0 0 1 -4 5 -2
%     0 0 0 0 0 0 0 0 0 0 0 0 1 -2 1 ]

A= load("A.txt");
rows= size(A,1);

for i=1:rows
    A(i,i)= A(i,i) +9;
end

[L,U, counter1] = LU_calc(A); % Calling LU_calc fn for finding L and U
 matrix for A
X= zeros (rows, rows);
counter2=0;
for i=1: rows
    B= zeros(rows,1);
    B(i)=1;
    [X(:,i),count]= inverse_calc(L,U,B); % Calling inverse_calc fn for
 getting vector x(i)
    counter2= counter2+count;
end
disp("Inverse of matrix A is: ");
disp(X);
fprintf('Number of operation = %i\n', counter1+counter2);
```

Inverse of matrix A is:
Columns 1 through 7

0.0591	0.0158	-0.0000	-0.0011	-0.0003	0.0000	0.0000
0.0158	0.0760	0.0192	-0.0003	-0.0015	-0.0004	0.0000
-0.0000	0.0192	0.0770	0.0192	-0.0004	-0.0015	-0.0004
-0.0011	-0.0003	0.0192	0.0770	0.0192	-0.0004	-0.0015
-0.0003	-0.0015	-0.0004	0.0192	0.0770	0.0192	-0.0004
0.0000	-0.0004	-0.0015	-0.0004	0.0192	0.0770	0.0192

```
    0.0000    0.0000   -0.0004   -0.0015   -0.0004    0.0192    0.0770
    0.0000    0.0000    0.0000   -0.0004   -0.0015   -0.0004    0.0192
   -0.0000    0.0000    0.0000    0.0000   -0.0004   -0.0015   -0.0004
   -0.0000   -0.0000    0.0000    0.0000    0.0000   -0.0004   -0.0015
   -0.0000   -0.0000   -0.0000    0.0000    0.0000    0.0000   -0.0004
    0.0000   -0.0000   -0.0000   -0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000
   -0.0000    0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000

  Columns 8 through 14

    0.0000   -0.0000   -0.0000   -0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000
   -0.0004    0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000
   -0.0015   -0.0004    0.0000    0.0000    0.0000   -0.0000   -0.0000
   -0.0004   -0.0015   -0.0004    0.0000    0.0000    0.0000   -0.0000
    0.0192   -0.0004   -0.0015   -0.0004    0.0000    0.0000    0.0000
    0.0770    0.0192   -0.0004   -0.0015   -0.0004    0.0000    0.0000
    0.0192    0.0770    0.0192   -0.0004   -0.0015   -0.0004    0.0000
   -0.0004    0.0192    0.0770    0.0192   -0.0004   -0.0015   -0.0004
   -0.0015   -0.0004    0.0192    0.0770    0.0192   -0.0004   -0.0015
   -0.0004   -0.0015   -0.0004    0.0192    0.0770    0.0193   -0.0003
    0.0000   -0.0004   -0.0015   -0.0004    0.0193    0.0775    0.0202
    0.0000    0.0000   -0.0004   -0.0015   -0.0003    0.0202    0.0792
    0.0000    0.0000    0.0001   -0.0003   -0.0020   -0.0037    0.0138

  Column 15

   -0.0000
    0.0000
    0.0000
    0.0000
   -0.0000
   -0.0000
   -0.0000
    0.0000
    0.0000
    0.0001
   -0.0003
   -0.0020
   -0.0037
    0.0138
    0.1031

Number of operation = 8870
```

Published with MATLAB® R2021a

```
function [L, U, counter]= LU_calc(A)
% Length of various matrices
size_A= size(A,1);
L= eye(size_A); % Initialized as identity matrix
counter=0;
for i= 1 : size_A
    % Pivot and largest diagonal element Condition
    diagonal_max=A(i,:);
    max_row=i;
    for k= i+1 : size_A
        if(abs(A(k,i)) > diagonal_max(i)) % Condition
            diagonal_max=A(k,:);
            max_row=k;
        end
    end
    A(max_row,:)=A(i,:); % Updating pivot value to max in column
    A(i,:)=diagonal_max;

    % Gauss-elimination method
    if abs(A(i,i)) > 1e-4 % Condition to ensure no operation is done
 on NULL element
        for j=i+1 : size_A
            factor= A(j,i)./A(i,i); % calculating factor
            counter= counter+1;
            L(j,i)= factor; % Updating L matrix
            A(j,:)=A(j,:)-factor.*A(i,:); % updating subsequent rows
            counter= counter+ 2.*(size_A -i +1);

        end
    end
end
U= A; % Modified A matrix is subsequently the required U matrix
end
```

Published with MATLAB® R2021a

```
function [x, counter]= inverse_calc(L,U,B)
rows= size(L,1);
d = zeros(rows,1); % Initialize d column vector
counter=0;
for i=1:rows
    B(i);
    temp= 0;
    for j=1:i-1
        temp= temp + L(i,j).*d(j);
    end
    counter= counter + 2.*(i-1) -1;
    d(i)= B(i)-temp; % Forward substitution formula
    counter= counter+1;
end

x= zeros(rows,1); % Initialize x column vector
for i=rows:-1:1
    temp= 0;
    for j=i+1:rows
        temp= temp + U(i,j).*x(j);
    end
    counter= counter+ 2.*(rows-i) -1;
    x(i)= (d(i)-temp)./U(i,i); % Backsubstitution formula
    counter= counter +2;
end
end
```

Published with MATLAB® R2021a