Date / /
Page No.

CL249 – Computational methods Lab
Assignment 2 : Solution of system of linear eqⁿ

Submitted by :

AYUSHMAN CHOUDHARY
(200020039)

Q1. Solve the given Ax = B problem using Gauss-Elimination with pivoting and diagonal dominant part. Put a counter in the program to count the total number of operations done (+, −, *, /). Finally, plot the value of x using plot(x).

Q2. Write two modular code. In main code, read matrix A from data file and call Gauss elimination code. In that code, matrix A and B are taken as input and x is provided as output. Gauss elimination code should have portion of pivoting and largest diagonal element.

METHOD : GAUSS ELIMINATION

Let's assume we have the given problem:

$$Ax = B$$

which on expansion appears as:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & \\ a_{21} & \ddots & & \\ \vdots & & \ddots & \\ \vdots & & & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

To proceed to solve using backsubstitution method, we need to convert matrix A into it Row Echelon Form (REF) i.e.

(i) The non zero rows of A precede the zero rows.

(ii) If A has r non zero rows, and the pivot in row 1 appears in col k_1, in row 2 appears in col k_2, and soon, then $k_1 < k_2 < \cdots k_r$.

$$\Rightarrow \begin{bmatrix} a_{11} & a_{12} & \cdots & \cdots \\ 0 & a_{22}' & \cdots & \cdots \\ 0 & 0 & \ddots & \\ \vdots & \vdots & 0 & \ddots \\ \vdots & \vdots & 0 & \\ 0 & 0 & & a_{nn}' \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1' \\ b_2' \\ \vdots \\ b_n' \end{bmatrix}$$

To achieve this form of A, we can perform row operations.

For eg. to make $a_{21} = 0$

perform $R_2 = R_2 - (\text{factor}) * R_1$

where $\text{factor} = \dfrac{a_{21}}{a_{11}}$

Carrying out this process for all the rows would give us REF of A.

PIVOTING

While calculating factor, we need to make sure that the denominator doesn't equals to 0. To do this we ensure that

$$\cancel{a_{ii} > a_{ji} \quad \forall j \geq i+1}$$

$$a_{ii} \neq 0$$

else we replace the row with another row where $a_{ji} \neq 0$.

Largest diagonal element

To ensure that there is minimum error in calculation,

$$a_{ii} > a_{ji} \quad \forall j \geq i+1$$

Now, using back substitution we can find x as

$$\Rightarrow a_{mn} \cdot x_m = b_m'$$

$$x_m = \frac{b_m'}{a_{mn}}$$

$$\Rightarrow a_{n-1\,n-1} \cdot x_{n-1} + a_{n-1\,n} x_n = b_{n-1}'$$

$$\Rightarrow x_{n-1} = \left(\frac{b_{n-1}' - a_{n-1\,n} x_n}{a_{n-1\,n-1}}\right)$$

$$\vdots$$

Therefore, we can find x vector that is the solution to

$$Ax = B.$$

Date
Page No.

PSEUDOCODE

Gauss Elimination

1. Start

2. Take user input A, B.

3. Y = [A B]

4. counter = 0

5. For i = 1 to row(Y):

% Pivot and largest diagonal Element.

```
diagonal_max = Y(i,:)
max_row = i
for k = i+1 to row(Y):
    if |Y(k,i)| > diagonal_max :
        update diagonal_max & max_row
    end if
end for
update Y(i,:)
```

% Gauss Elimination method

```
if |Y(i,i)| > ε
    For j = i+1 to row(Y)
```

```
            factor = Y(j,i) / Y(i,i)

            Y(j,:) = Y(j,:) - (factor)* Y(i,:)
            update counter

        end FOR

    end IF

end FOR
```

6. % Backsubstitution

```
X = zeros (1, col(A))

for i = row(Y) : -1 : 1
        temp = sum(Y(i, i+1 : col(Y)-1) *
                        x(i+1 : size col(X)))

        x(i) = (Y(i, col(Y)) - temp) / Y(i,i)

        update counter

END for.
```

Comments & Remarks

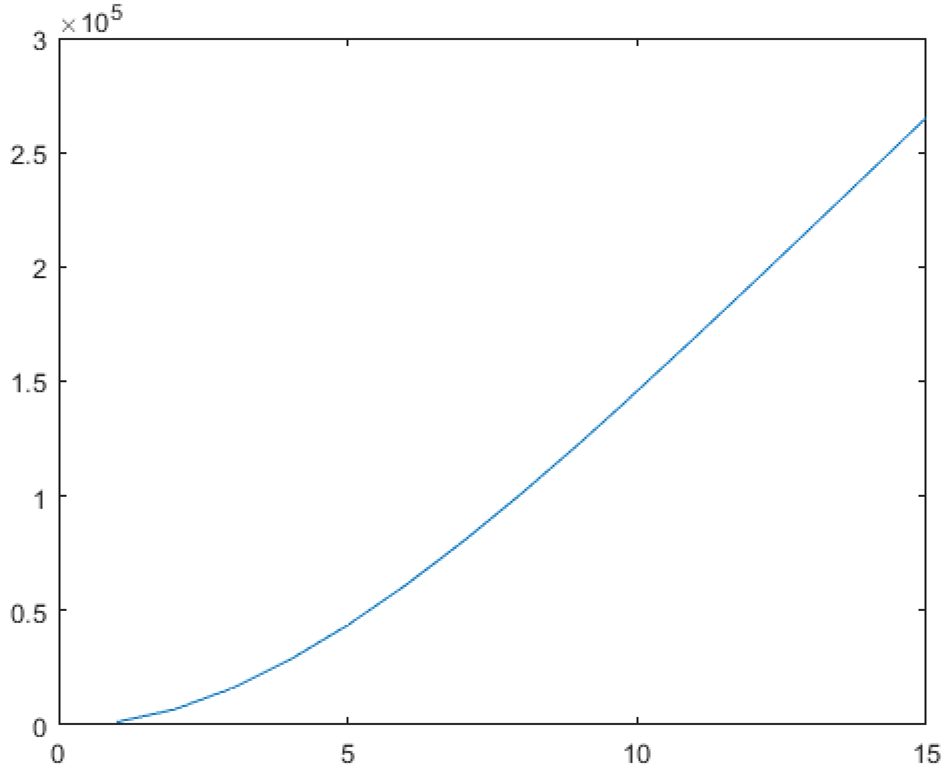
- Taking $Y = [A\ B]$ simplifies the process as there is no need update A and B separately.

- Doing vector multiplication reduces the need for another for loop.

- Gauss Elimination is a better method than Cramer's Rule due to its better time complexity.

```
clear all;
clc

% Solving Ax=B using Gauss Elimination method and Back substitution
A= load("A.txt"); % loading A matrix from data file

B= 41*ones(15,1); % B matrix is a constant matrix

X = Gauss_elimination(A,B); % Calling Gauss_Elimination function for
 solving Ax=B

plot(X);
```

Number of operation = 2570



Published with MATLAB® R2021a

```
function X= Gauss_elimination(A,B)
% Length of various matrices
len_B= size(B);
row_A= size(A,1);
col_A= size(A,2);
Y=[A B]; % Y= [A|B] format to minimize separate operations on A and B.
row_Y= size(Y,1);
col_Y= size(Y,2);

counter=0;
for i= 1 : row_Y
    % Pivot and largest diagonal element Condition
    diagonal_max=Y(i,:);
    max_row=i;
    for k= i+1 : row_Y
        if(abs(Y(k,i)) > diagonal_max(i)) % Condition
            diagonal_max=Y(k,:);
            max_row=k;
        end
    end
    Y(max_row,:)=Y(i,:); % Updating pivot value to max in column
    Y(i,:)=diagonal_max;

    % Gauss-elimination method
    if abs(Y(i,i)) > 1e-4 % Condition to ensure no operation is done
 on NULL element
        for j=i+1 : row_Y

            factor= Y(j,i)./Y(i,i); % calculating factor
            counter= counter+1;
            Y(j,:)=Y(j,:)-factor.*Y(i,:); % updating subsequent rows
            counter = counter + 2.*(col_Y - i);
        end
    end
end
% Back Substitution
X = zeros(1,col_A); % Initializing X vector to 0
col_X= size(X,2);
for i=row_Y:-1:1
    temp= sum(Y(i,i+1:col_Y-1).*X(i+1:col_X));
    counter= counter + 2.*(col_X -i) -1;
    X(i)= (Y(i,col_Y)-temp)./Y(i,i); % Backsubstitution formula
    counter= counter +2;
end
fprintf('Number of operation = %i\n', counter);
end
```

Published with MATLAB® R2021a