Gaussian Elimination Method with Backward Substitution Using Matlab

Huda Alsaud

King Saud University

The main purpose of these slides is to demonstrate how to write a function m-file that will solve an arbitrary system $(Ax = b)$ of N linear equations in N unknowns $x_i, i = 1 : N$ using the Gaussian Elimination algorithm as covered in class. The MATLAB program of the Gaussian Elimination algorithm can be done in various ways. However, since these slides were prepared for students how didn't learn MATLAB before, we will present some MATLAB statements which will be used in the program, but we limit the selection to the material which is needed later and for more details we refer to the references [1] and [2].

The program will be as follows

- A function m-file, with inputs $A = (a_{ij})_{N\times N}$ and $b = (b_j)_{N\times 1}$ and output the solution vector, $x = (x_i)_{N\times 1}$.
- Step 1 For $j = 1 : (N - 1)$ do steps 2-3
- Step 2 If $a_{jj} = 0$ then find the smallest integer $j < p \leqslant N$ and $a_{pj} \neq 0$ and then $E_j \leftrightarrow E_p$. If no integer p can be found then output (The system has no unique solution).
- Step 3 For $i = j + 1 : N$

$$m = a_{ij}/a_{jj} \quad \text{preform} \quad E_i - mE_j \to E_i.$$

- Step 4 If $a_{NN} = 0$ then output (The system has no unique solution).
- Step 5 Set $x_N = b_N/a_{NN}$.
- Step 6 For $i = N - 1 : 1$

$$x_i = (b_i - \sum_{j=i+1}^{N} a_{ij}x_j)/a_{ii}.$$

- Step 7 output $x_1, \cdots, x_N$.

We will discuss the following

(1) Vectors and Matrices.

(2) For Statement.

(3) If Statement.

(4) Functions that return more than one value.

(5) Create a M-file to calculate Gaussian Elimination Method with Backward Substitution.

(6) Homework.

Vectors and Matrices

Create Row Vector.

There are several ways to create row vector variables. The most direct way is to put the values in square brackets, separated by either space or commas.

Example

$>> v = [1\ 4\ \ -2\ 0]$

or

$>> v = [1, 4, -2, 0]$

The elements in a vector are numbered sequentially each element number is called the index.

Example

```
   1  2  3  4
v=[1 4 -2 0]
```

The nth element in the vector is $v(n)$.

Example

```
>>v(3)
ans = -2
```

Create Matrix Variables

The values within a row are separated by either space or commas, and the different rows are separated by semicolons.

Example

```
   1  2  3  4
v=[1 4 -2 0]
```

The nth element in the vector is $v(n)$.

Example

```
>>v(3)
ans = -2
```

Create Matrix Variables

The values within a row are separated by either space or commas, and the different rows are separated by semicolons.

Example

$>> A = [1\ 4\ -8; 3\ 0\ -5]$

A=

$$\begin{matrix} 1 & 4 & -8 \\ 3 & 0 & -5 \end{matrix}$$

To refer to matrix elements the row and then the column subscripts are given in parentheses.

Example

$>> A(1,2)$

ans=

$$4$$

Example

$>> A = [1\ 4\ -8; 3\ 0\ -5]$

A=

$$\begin{matrix} 1 & 4 & -8 \\ 3 & 0 & -5 \end{matrix}$$

To refer to matrix elements the row and then the column subscripts are given in parentheses.

Example

$>> A(1,2)$

ans=

$$4$$

To refer to entire row use a colon for the column subscript,

Example

$>> A(1,:)$

ans=

1 4 -8

This refer to the entire second column,

Example

$>> A(:,2)$

ans= 4

0

To refer to entire row use a colon for the column subscript,

Example

$>> A(1,:)$

ans=

1 4 -8

This refer to the entire second column,

Example

$>> A(:,2)$

ans= 4

0

Dimensions

The *length* and *size* functions in MATLAB are used to find dimensions of vectors and matrices.

Example

$>> v = [1\ 4\ -2\ 0]$

v=

1 4 -2 0

$>> \text{length}(v)$

ans=

4

Example

$>> A = [1 \quad 4 \quad -8;\ 3 \quad 0 \quad -5]$

$A =$

$$\begin{matrix} 1 & 4 & -8 \\ 3 & 0 & -5 \end{matrix}$$

$>> \text{size}(A)$

ans=

2 3

For Statement

The for statement, or the for loop, is used when it is necessary to repeat statements in a script or a function, and when it is known ahead of time how many times the statements will be repeated.

The general form of the For loop is

```
For i=range
        action
end
```

Example: Calculate the sum $\sum_{i=1}^{n} i$

```
s=0
for  i=1:n
s=s+i
end
```

For Statement

The for statement, or the for loop, is used when it is necessary to repeat statements in a script or a function, and when it is known ahead of time how many times the statements will be repeated.

The general form of the For loop is

```
For i=range
        action
end
```

Example: Calculate the sum $\sum_{i=1}^{n} i$

```
s=0
for  i=1:n
s=s+i
end
```

Breaking from a Loop.

Sometimes you may want MATLAB to jump out of a for loop, for example if a certain condition is met. Inside the loop, you can use the command *break* to tell MATLAB to stop running the loop and skip to the next line after the end of the loop. For example, to compute $\sum_{N=1}^{100} \frac{1}{N^4}$ and stop only when the terms become so small compared with the machine precision that the numerical sum stops changing we use the command *break* .

Example: Compute $\sum_{N=1}^{100} \frac{1}{N^4}$

```
s=0
for N=1:100
    a=s
    s=s+N^ (-4)
        if s=a
            break
        end
end
```

To stop executing of M-file, without running any further commands, use the command *return*.

If Statement

The if statement choose whether a statement, or group of statements, is executed or not.

The general form of the if statement follows

If condition

    action

end

Example

if $k < 10$

    $k = k * 4$

end

If Statement

The if statement choose whether a statement, or group of statements, is executed or not.

The general form of the if statement follows

If condition

    action

end

Example

if $k < 10$

    $k = k * 4$

end

When the if statement is executed, the condition is evaluated. If the value of the condition is true the action will be executed, if not the action will not be executed.

To choose between two statements use if-else statement.

The general form is

```
if      condition
      action 1
else
      action 2
end
```

To choose from among more than two actions use elseif.

The general form is

```
if      condition 1
     action 1
elseif   condition 2
     action 2
else
     action 3
end
```

Example

if $k > 1$

fprintf('k is greater than 1')

elseif $k < 1$

fprintf('k is less than 1')

else

fprintf('k is equal to 1')

end

Functions that return more than one value

Function that return more than one value must have more than one output argument in the function header in square brackets.

The general form looks like the following

function [output argument]=functionname(input argument)

    statement here

end

Example

$function[area, circum] = areacirc(rad)$

$area = pi * rad\hat{}2;$

$circum = 2 * pi * rad;$

end

Functions that return more than one value

Function that return more than one value must have more than one output argument in the function header in square brackets.

The general form looks like the following

function [output argument]=functionname(input argument)

statement here

end

Example

$function[area, circum] = areacirc(rad)$

$area = pi * rad\hat{}2;$

$circum = 2 * pi * rad;$

end

Example

$function[r] = f(s)$

$r(1) = sqrt(s(1)\hat{}2 + s(2)\hat{}2);$

$r(2) = s(2)/s(1)$

end

The above function takes as argument a vector of 2 components and returns a vector of 2 components.

Notice that MATLAB requires a double equals sign $==$ to test for equality, a single equals is reserved for the assignment of value to variables.

Create a M-file to calculate Gaussian Elimination Method

```
function [x]=Guss(A,b)
N=length(b);
% Gaussian elimination
for j=1:(N-1)
    if A(j,j)==0%Check if a(i,i)=0 or not
        for k=j+1:N
            if A(k,j)~=0
                [h]=A(k,:);
                A(k,:)=A(j,:);
                A(j,:)=[h];
                t=b(k);
                b(k)=b(j);
                b(j)=t;
                break
            elseif k==N
                    fprintf('The system has no unique solution')
                    return
            end
        end
    end
    for i=j+1:N
        m=A(i,j)/A(j,j);
        A(i,:)=A(i,:)-m*A(j,:);
        b(i)=b(i)-m*b(j);
    end
    end
    if A(N,N)==0%check if a(n,n)=0 or not
        fprintf('The system has no solution')
        return
    else
%back substitution
x(N)=b(N)/A(N,N);
for i=N-1:-1:1
    h=0;
    for j=(i+1):N
        h=h+A(i,j)*x(j);
    end
    x(i)=(b(i)-h)/A(i,i);
end

    end
```

Guss Ln 34 Col 9

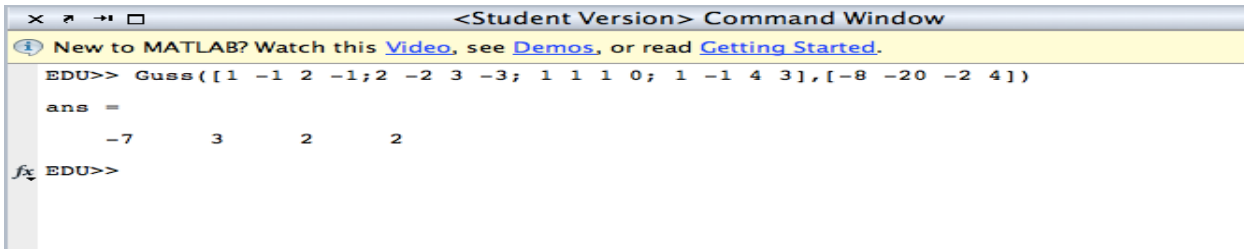
Example

Solve the linear system

$$\begin{aligned} x_1 - x_2 + 2x_3 - x_4 &= -8 \\ 2x_1 - 2x_2 + 3x_3 - 3x_4 &= -20 \\ x_1 + x_2 + x_3 &= -2 \\ x_1 - x_2 + 4x_3 + 3x_4 &= 4 \end{aligned}$$

using the M-file *Guss.m*.

Solution



Homework

Q1: Give two examples of linear systems such that one of them has no unique solution and the other has unique solution and then solve them using Guss.m file.

Q2: Write a system of linear equations where the solution of the system will be your last four digits of your university number, and then solve the system using Guss.m file.

Q3: Change the file Guss.m so that whenever one of the pivots $a_{kk}^{(k)}$ is zero print an error message and stop the program execution.

References


B.R. Hunt, R.L. Lipsman, and J.M. Rosenberg. A Guide to MATLAB, for beginners and experienced users. Cambridge University Press, 2001.


Stormy Attaway. MATLAB: A Practical Introduction to Programming and Problem Solving. Elsevier Inc, 2012.