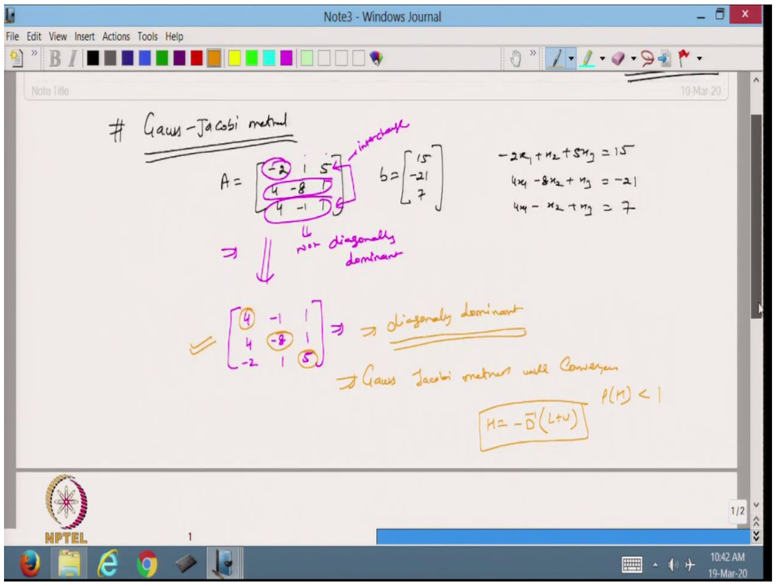
# Scientific Computing Using MATLAB
**Professor Vivek Aggarwal**
**Department of Mathematics**
**Indian Institute of Technology, Delhi/DTU**
**Lecture - 26**
**MATLAB Code for Gauss-Jacobi Method**

Hello, viewers. Welcome back to the course on Scientific Computing and we are using MATLAB. So in the previous class, we have discussed the Gauss-Seidel method and the Gauss-Jacobi and their convergence, the necessary condition for convergence. So today we will make the MATLAB code for that one.

(Refer Slide Time: 00:43)



# Gauss-Jacobi method

$$A = \begin{bmatrix} -2 & 1 & 5 \\ 4 & -8 & 1 \\ 4 & -1 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 15 \\ -21 \\ 7 \end{bmatrix}$$

interchange

$$-2x_1 + x_2 + 5x_3 = 15$$
$$4x_1 - 8x_2 + x_3 = -21$$
$$4x_1 - x_2 + x_3 = 7$$

$\Rightarrow$ not diagonally dominant

$$\begin{bmatrix} 4 & -1 & 1 \\ 4 & -8 & 1 \\ -2 & 1 & 5 \end{bmatrix} \Rightarrow \Rightarrow \text{diagonally dominant}$$

$\Rightarrow$ Gauss Jacobi method will Converge

$$\boxed{H = -D^{-1}(L+U)} \quad \rho(H) < 1$$

But before that, I will, so today I will make the MATLAB code for Gauss-Jacobi method and in that method, we will be going to solve this system. So I am going to take this matrix as -2 1 5, 4 -8 1, and then 4 -1 1; so that is my A and B, I am going to take 15 -21 and 7.

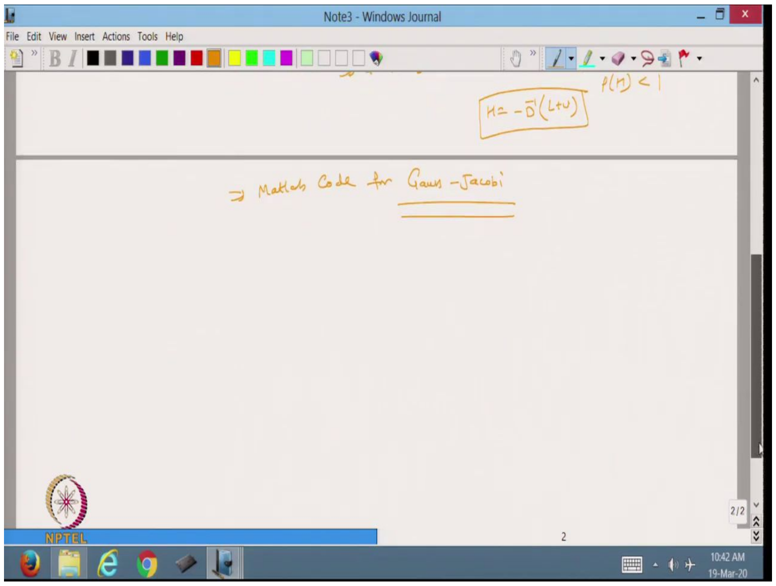
So this is a -2 1 5, 4 -8 1, 4 minus 1 minus 1, and this is the right-hand side. So if I take this one, basically my system is this. So minus 2x1+x2+5x3=15, 4x1-8x2+x3=-21, and 4x1-x2+x3=7. So in this case I want to see whether this matrix is diagonally dominant or not.

So if you, from here I can see that this part is even less than some of this, so it is not diagonally dominant or even if I change this matrix into the this form, I will take it interchanging the rows, so maybe I will change this row here but because in this case also, this is ok, it is diagonally dominant but it is not diagonally dominant.

So suppose I take 4 here, -1 1, and then 4 -8 1, and then -2 1 5. So I have interchanged the first and the, so this and this row, I have interchanged. So from here, I can change this one, and now if I try to find out, I can see that this sum is greater than this, 8 is also greater than this number, and 5 is also greater than this number. So this matrix is diagonally dominant.

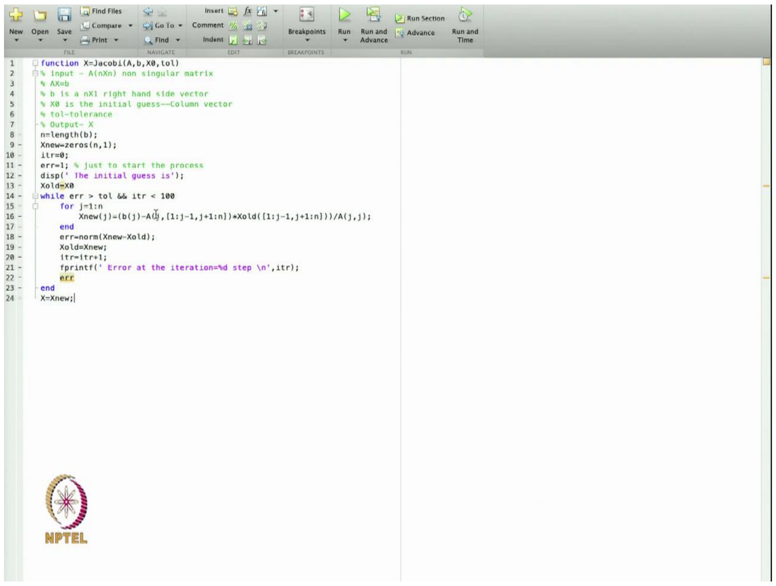
So if I take this matrix instead of this matrix, then my Gauss-Jacobi method will converge, definitely converge. So the Gauss-Jacobi method will converge. So this one I can see from here and if this will converge then definitely, I told that the eigenvalues will be less than 1, the spectral radius will be less than 1 for the Gauss-Jacobi method, and where my $H = -D^{-1}(L+U)$. So that is my corresponding matrix.

(Refer Slide Time: 04:41)


Note3 - Windows Journal
$\rho(H) < 1$
$\boxed{H = -D^{-1}(L+U)}$
$\Rightarrow$ Matlab Code for Gauss-Jacobi
NPTEL

So now we are going to deal with MATLAB code for Gauss-Jacobi. So let us go to the MATLAB.

(Refer Slide Time: 04:59)



```
function X=Jacobi(A,b,X0,tol)
% input - A(nXn) non singular matrix
% AX=b
% b is a nX1 right hand side vector
% X0 is the initial guess--Column vector
% tol-tolerance
% Output- X
n=length(b);
Xnew=zeros(n,1);
itr=0;
err=1; % just to start the process
disp(' The initial guess is');
Xold=X0
while err > tol && itr < 100
    for j=1:n
        Xnew(j)=(b(j)-A(j,[1:j-1,j+1:n])*Xold([1:j-1,j+1:n]))/A(j,j);
    end
    err=norm(Xnew-Xold);
    Xold=Xnew;
    itr=itr+1;
    fprintf(' Error at the iteration=%d step \n',itr);
    err
end
X=Xnew;
```

So I have made this code already for you. So this is the function I am defining, Jacobi, I am giving the name Jacobi. So I have to input what is that A value, the matrix; the right-hand side vector b; and this is, X0 is the initial approximation I am going to start with; and this is the tolerance value I am going to give.

Because this is an iterative process, so I need to give this initial condition and the tolerance and the output will be the solution. So I have written there that input is a matrix that should be a non-singular matrix and my AX should be, this is my system; b is my right-hand side vector; and X0 is the initial guess, the column vector; and this is the tolerance. So I am passing all the column vectors here.

Now what I will do? I take the length of b, so that is my n. So this is n I have defined because it should be a square matrix. Now what I do is take a new vector Xnew with all values 0 and it should be a column vector, so this is n and 1. Now, I start with the iteration. So iteration is 0 and for the, just to start with this process, I just take the error equal to 1 because, later on, I will tell you why I am writing this one.

Now I will display that the initial guess is X0, so X0 will be the initial guess and I am putting this value in the X old vector. Now I do the iterative process, so this is my while-loop. So I, in the while-loop I say that if error is greater than tolerance and the iteration is less than 100, so I am keeping the maximum iteration here that is 100. So if both the conditions are satisfied, only then it will go to this loop. So that is why I am putting the error=1 here so that both the condition will be satisfied and then while-loop will start.

So just to start the loop, I will give the value error=1. Now I will go to the while-loop. Now, this is my for-loop, so I will go from 1 to n, and Xnew means the left-hand side of the iterative process should be equal
$Xnew(j) = (b(j) - A(j, [1 : j - 1, j + 1 : n]) * Xold([1 : j - 1, j + 1 : n]))/A(j, j)$
. I told you that bj is the right-hand side element -A with the same row, so that is the jth row and all the columns, except the jth column. So I am putting A 1 to j-1 and j+1 to n. So only leaving the j j column from here.

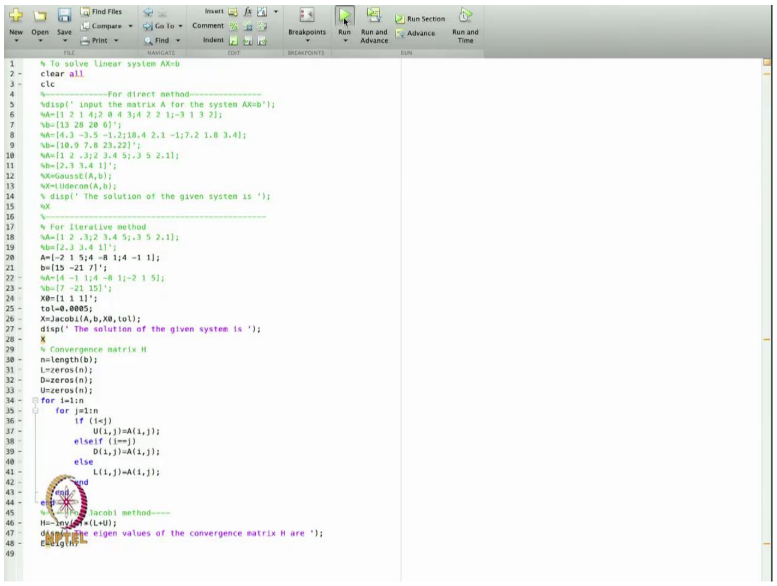
So this multiply by the X old, so X old will be the same as the same argument, this one, this indexing. So the same indexing X old should be 1 to j-1 and j+1 to n because we have to divide this number by this. So I am taking all these numbers here and then I am dividing by the diagonal element and that is A(j,j).

So in this case, what I am doing, I am giving the old value of the X and I am getting the new value of X. So after this for-loop, I will go to the error. So error I am finding
$error = norm(xnew - xold)$, so the norm means that it should be, it is a L2 norm. And I am taking the L2 norm, so absolute value, and then giving the error. So that is my error value, so the error value is there. Now, I then no need to write the absolute here, because the norm will be positive.

Now I am putting X new transferring to the X old, $Xold = Xnew$, iteration is increased by 1 , $itr = itr + 1$, and I am writing the error at the iteration, at this iteration step is this one. Now, I will go to the while-loop again, doing the same process again and again until this condition is satisfied.

So now, suppose my error is reduced and this error becomes less than tolerance. So I will stop here because in this case, this condition not satisfying end condition and it will stop or my iterations are too large and is going more than 100, so in that case, also, it will stop and the answer I will get, so Xnew, whatever the value of Xnew is there, that will be go to the X and this will be the output of my code. So that is the Gauss-Jacobi method. So it is very small code for this one.

(Refer Slide Time: 09:49)



```
% To solve linear system AX=b
clear all
clc
%------------For direct method-------------
%disp(' input the matrix A for the system AX=b');
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8 3.4];
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given system is ');
%X
%--------------------------------------------
% For Iterative method
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
A=[-2 1 5;4 -8 1;4 -1 1];
b=[15 -21 7]';
%A=[4 -1 1;4 -8 1;-2 1 5];
%b=[7 -21 15]';
X0=[1 1 1]';
tol=0.0005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given system is ');
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
    for j=1:n
        if (i<j)
            U(i,j)=A(i,j);
        elseif (i==j)
            D(i,j)=A(i,j);
        else
            L(i,j)=A(i,j);
        end
    end
end
%----For Jacobi method----
H=-inv(D)*(L+U);
disp(' The eigen values of the convergence matrix H are ');
E=eig(H)
```

Now I will go to the main code from where I will call this function. So if you remember, last time I made this linear system code, that will solve all the linear systems. So now we have to do the iterative process, so that is why I have commented on the previous one that we have used for the direct method. So this is my clear all and that is clc. Now I will start with the iterative process.

So now, this is the matrix I have started with -2 1 5, 4 -8 1, and 4 -1 1; and the right-hand side vector is the column vector. So this is what I just started on the slides. So now I am, so this is my initial value I am passing. So I am, for example, I am taking 1 1 1, the tolerance I am defining is 0.0005.

So it means that is half into $10^{-3}$. So that is my tolerance and from here, I am calling Jacobi method A, b, X0, and tolerance. I am passing my A, b, X0, and tolerance, and then from there, I am displaying that whatever the solution I will receive, that is a solution.

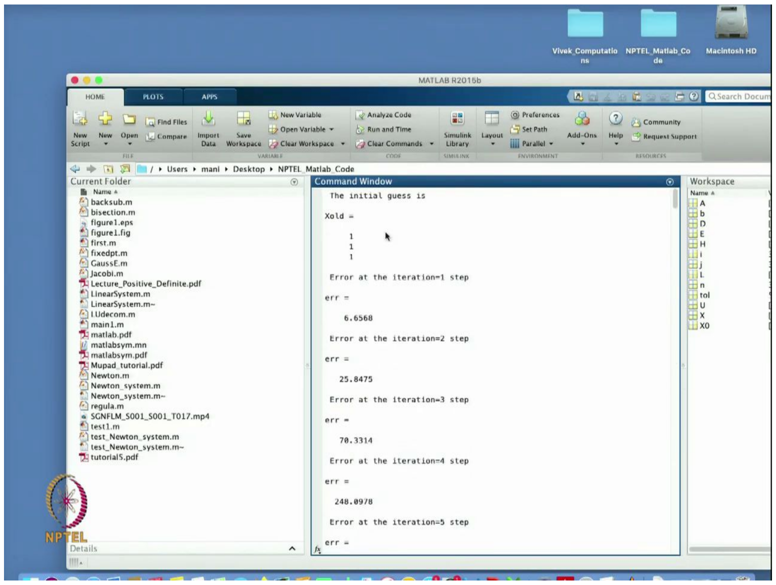
Now I have to find out the convergence matrix. So for the convergence matrix, this is the convergence matrix H, I am writing; n is the length of the b; L is the matrix, n*n matrix; D is n*n matrix; U is n*n matrix. And I am defining with the 0 because I know that in the diagonal elements except the diagonal, all element zeros, in L also, most of the elements are zero, U also.

So now I am writing this for-loop. So for i is equal to 1 to n, for j is equal to 1 to n when i is less than j, so when i is less than j, it means I am talking about the upper triangular matrix. So my $U(i,j) = A(i,j)$. When i is equal to j, it means I am talking about the diagonal elements then it should be $D(i,j) = A(i,j)$, this one; else the lower triangular matrix, so $L(i,j) = A(i,j)$ and then I will end this if-loop, for, for; and from there I will get the matrix L, U, and D.

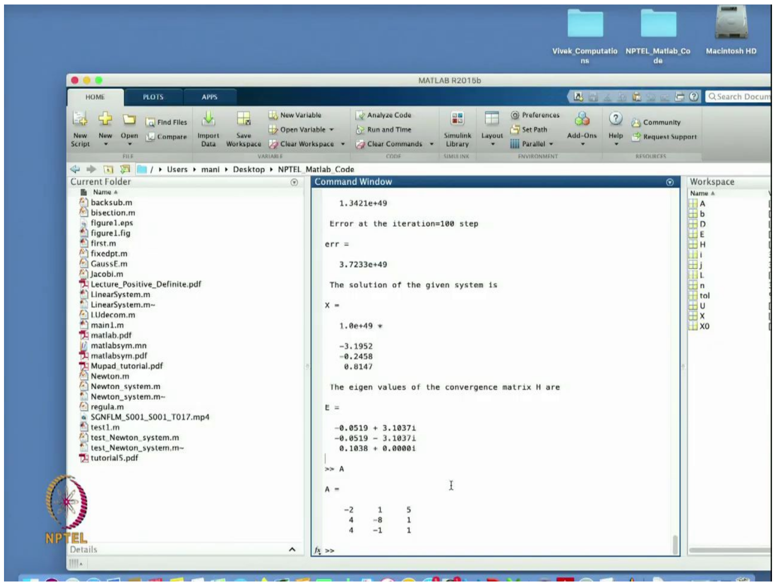
And then for the Jacobi method, I know that my convergence matrix is $-D^{-1}(L+U)$. So that in my method, that in my convergence matrix and then based on this one, I want to find the eigenvalues of this matrix. So this is the command in the MATLAB eig(H), so it will give the eigenvalues of the matrix.

And I am showing the display here that eigenvalues of the convergence matrix, H, are this value. So this is what I am doing. So let us run this code.

(Refer Slide Time: 13:07)



```
 The initial guess is

Xold =

     1
     1
     1

 Error at the iteration=1 step

err =

    6.6568

 Error at the iteration=2 step

err =

   25.8475

 Error at the iteration=3 step

err =

   70.3314

 Error at the iteration=4 step

err =

  248.0978

 Error at the iteration=5 step

err =
```



```
   1.3421e+49

 Error at the iteration=100 step

err =

   3.7233e+49

 The solution of the given system is

X =

   1.0e+49 *

   -3.1952
   -0.2458
    0.8147

 The eigen values of the convergence matrix H are

E =

  -0.0519 + 3.1037i
  -0.0519 - 3.1037i
   0.1038 + 0.0000i

>> A

A =

    -2     1     5
     4    -8     1
     4    -1     1

>>
```

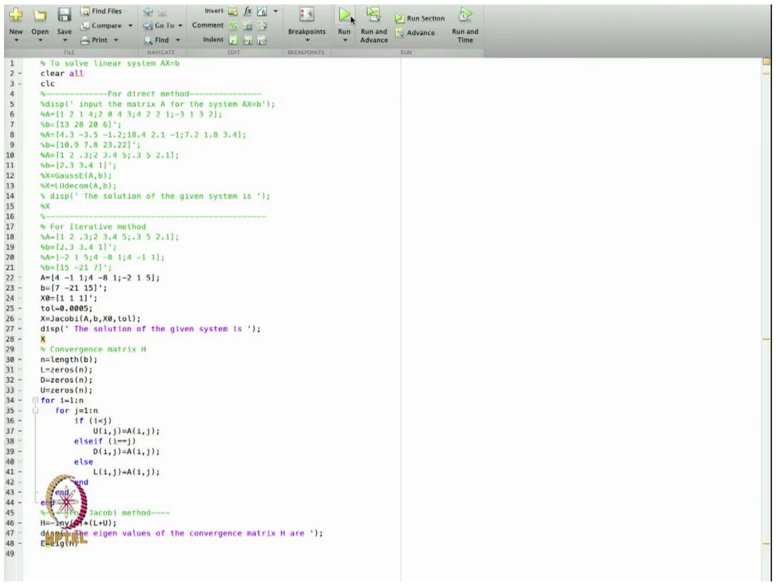
So I run this code, 1, and then, so this is the (MAT) solution I am getting. So I am starting with the 1, 1 and the error at the first iteration is very high, it is 6.65. Then as it is increasing, the error is going to increase and after some time, you see that error, it becomes a very large number.

It means that this process is not going to converge and maybe after a 100 iterations, this is my error. So it is infinite. So this process is not going to converge in this case and my X, the solution will be this one.

So this is, you know why it is happening, because the, whatever the matrix I have defined, so this is my matrix A. So I told you that this matrix is not diagonally dominant. So if the matrix is not diagonally dominant, that it may or may not converge. So it depends upon that, it may converge or it may not. So in this case, I can say that my Gauss-Jacobi method is not converging and this is the eigenvalues I am getting.

So magnitude is quite high here. If you see, this value is, imaginary part is 3, 3, and in this, the value is 0.1. So real eigenvalue is okay, it is 0.1 but the method is not converging, so I cannot talk about this one. So now if it is not converging, so let us do the interchanging of the rows. So let us do this one.

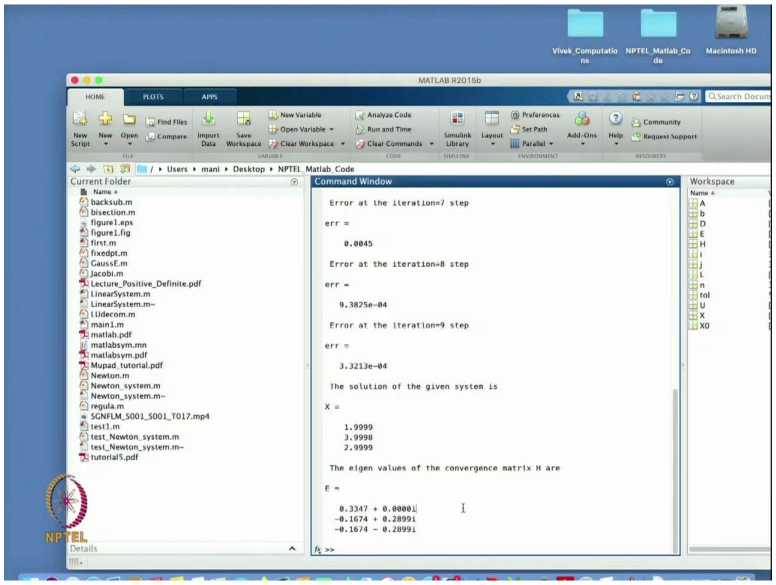
(Refer Slide Time: 14:46)



```
% To solve linear system AX=b
clear all
clc
%------------For direct method--------------
%disp(' input the matrix A for the system AX=b');
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8 3.4];
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given system is ');
%X
%---------------------------------------------
% For Iterative method
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%A=[-2 1 5;4 -8 1;4 -1 1];
%b=[15 -21 7]';
A=[4 -1 1;4 -8 1;-2 1 5];
b=[7 -21 15]';
X0=[1 1 1]';
tol=0.0005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given system is ');
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
    for j=1:n
        if (i<j)
            U(i,j)=A(i,j);
        elseif (i==j)
            D(i,j)=A(i,j);
        else
            L(i,j)=A(i,j);
        end
    end
end
%----For Jacobi method----
H=-inv(D)*(L+U);
disp(' The eigen values of the convergence matrix H are ');
E=eig(H)
```

Now I change the rows. Now, let us write this one. So now, you can see that 4 -1 -1, so that is okay, diagonally dominant. This is also diagonally dominant and this is also diagonally dominant. So if I change the interchange of the rows the right-hand side vector will also change and now I take the same process and I run the code.

(Refer Slide Time: 15:21)

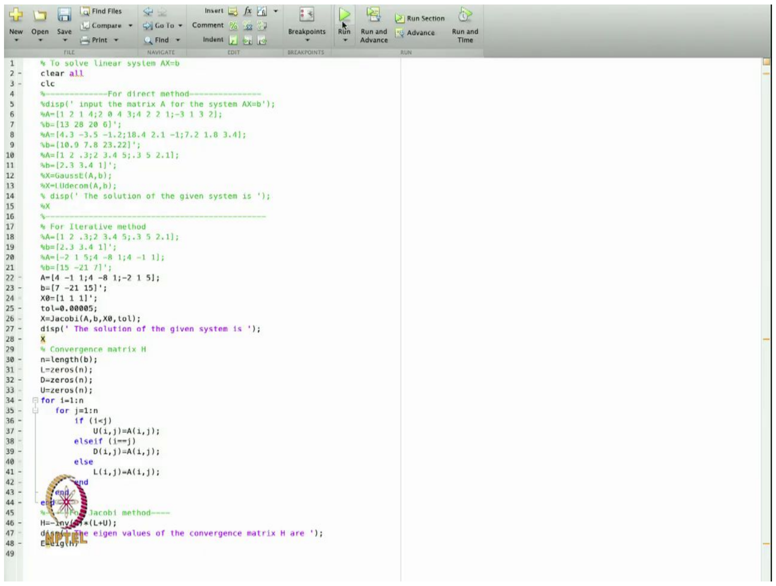


```
 Error at the iteration=7 step

err =

    0.0045

 Error at the iteration=8 step

err =

   9.3825e-04

 Error at the iteration=9 step

err =

   3.3213e-04

 The solution of the given system is

X =

    1.9999
    3.9998
    2.9999

 The eigen values of the convergence matrix H are

E =

   0.3347 + 0.0000i
  -0.1674 + 0.2899i
  -0.1674 - 0.2899i

>>
```

Now, you can see that my system, the error is after the ninth iteration, my error is this one and I get my solution. So it is 1.99, 3.99, 2.99 it means the solution is 2, 4, and 3. So my system is converging and if you see the eigenvalues, so these are the eigenvalues.

So real eigenvalue is 0.3347. So the maximum eigenvalue, in this case, the real is this one, and if I take the magnitude of the complex eigenvalue, so that is also less than 1. So because maximum eigenvalue we can talk about only in the real number. So my, this is the eigenvalue we will consider and its value is 0.3347.

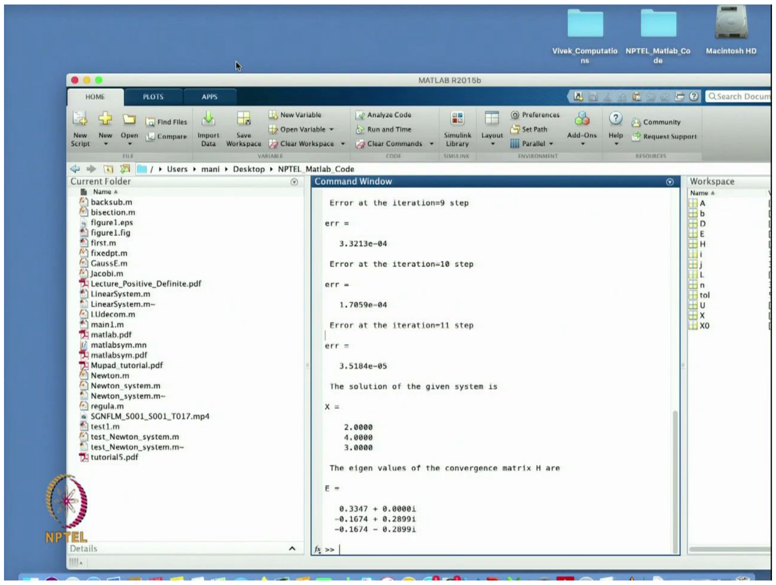
(Refer Slide Time: 16:12)



```
% To solve linear system AX=b
clear all
clc
%-------------For direct method--------------
%disp(' input the matrix A for the system AX=b');
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8 3.4];
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given system is ');
%X
%----------------------------------------------
% For Iterative method
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%A=[-2 1 5;4 -8 1;4 -1 1];
%b=[15 -21 7]';
A=[4 -1 1;4 -8 1;-2 1 5];
b=[7 -21 15]';
X0=[1 1 1]';
tol=0.00005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given system is ');
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
    for j=1:n
        if (i<j)
            U(i,j)=A(i,j);
        elseif (i==j)
            D(i,j)=A(i,j);
        else
            L(i,j)=A(i,j);
        end
    end
end
%----For Jacobi method----
H=-inv(D)*(L+U);
disp(' The eigen values of the convergence matrix H are ');
E=eig(H)
```

Now the same process, suppose I increase my tolerance and I want my accuracy up to four-digit, so let us run this code.

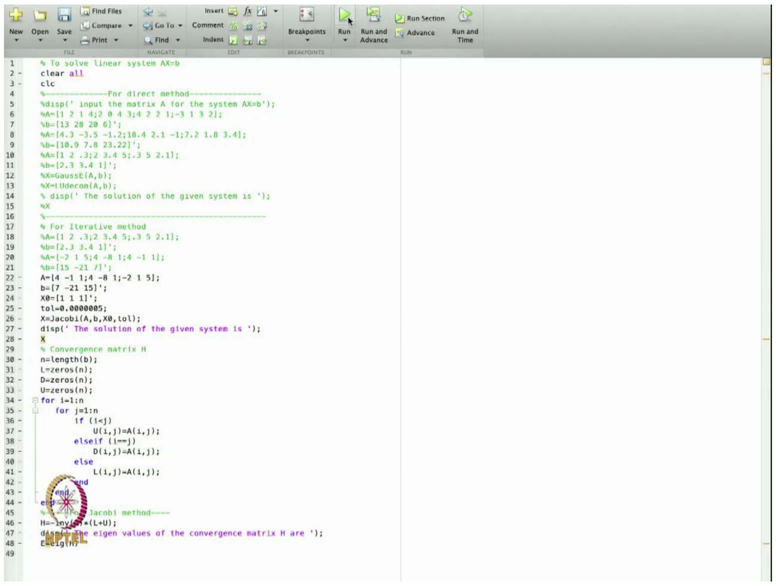
(Refer Slide Time: 16:23)



```
 Error at the iteration=9 step

err =

   3.3213e-04

 Error at the iteration=10 step

err =

   1.7059e-04

 Error at the iteration=11 step

err =

   3.5184e-05

 The solution of the given system is

X =

    2.0000
    4.0000
    3.0000

 The eigen values of the convergence matrix H are

E =

   0.3347 + 0.0000i
  -0.1674 + 0.2899i
  -0.1674 - 0.2899i

>>
```

Now you can see that the number of iterations becomes 11 and that is my solution. Now I am getting the exact solution, 2, 4, 3 after the 11 iteration. And the eigenvalue will be the same because eigenvalue is not going to change.

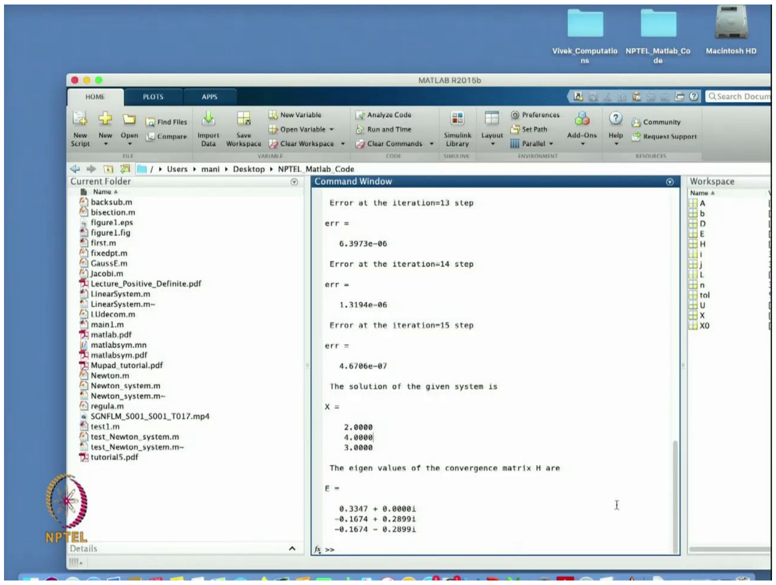
(Refer Slide Time: 16:41)



```
% To solve linear system AX=b
clear all
clc
%-------------For direct method--------------
%disp(' input the matrix A for the system AX=b');
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8 3.4];
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given system is ');
%X
%-------------------------------------------------
% For Iterative method
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%A=[-2 1 5;4 -8 1;4 -1 1];
%b=[15 -21 7]';
A=[4 -1 1;4 -8 1;-2 1 5];
b=[7 -21 15]';
X0=[1 1 1]';
tol=0.0000005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given system is ');
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
    for j=1:n
        if (i<j)
            U(i,j)=A(i,j);
        elseif (i==j)
            D(i,j)=A(i,j);
        else
            L(i,j)=A(i,j);
        end
    end
end
%----For Jacobi method----
H=-inv(D)*(L+U);
disp(' The eigen values of the convergence matrix H are ');
E=eig(H)
```

Now, maybe I can increase my tolerance and let us see what will happen.

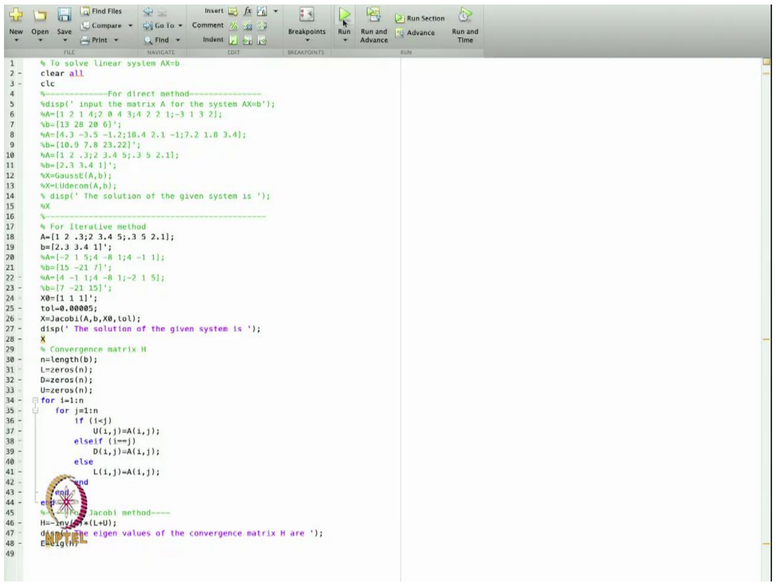
(Refer Slide Time: 16:53)



```
 Error at the iteration=13 step

err =

   6.3973e-06

 Error at the iteration=14 step

err =

   1.3194e-06

 Error at the iteration=15 step

err =

   4.6706e-07

 The solution of the given system is

X =

    2.0000
    4.0000
    3.0000

 The eigen values of the convergence matrix H are

E =

   0.3347 + 0.0000i
  -0.1674 + 0.2899i
  -0.1674 - 0.2899i

>>
```

So in this case, I get my 15 iteration and the solution I am getting is this one. So in this case, my matrix is diagonally dominant and my Gauss-Jacobi method is going to converge. So that is what we have done with the help of this MATLAB code. Now, maybe I can change some other matrix and let us see what will happen.

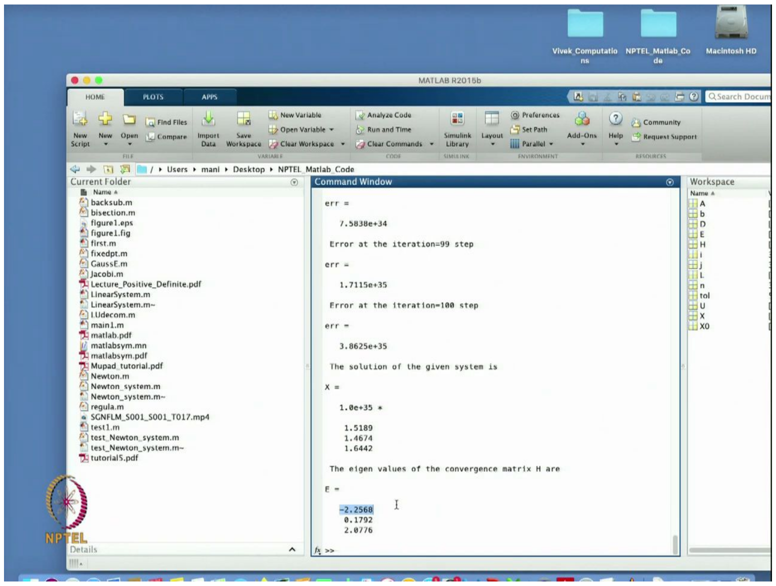
(Refer Slide Time: 17:27)



```
% To solve linear system AX=b
clear all
clc
%------------For direct method--------------
%disp(' input the matrix A for the system AX=b');
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8 3.4];
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given system is ');
%X
%--------------------------------------------
% For Iterative method
A=[1 2 .3;2 3.4 5;.3 5 2.1];
b=[2.3 3.4 1]';
%A=[-2 1 5;4 -8 1;4 -1 1];
%b=[15 -21 7]';
%A=[4 -1 1;4 -8 1;-2 1 5];
%b=[7 -21 15]';
X0=[1 1 1]';
tol=0.00005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given system is ');
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
    for j=1:n
        if (i<j)
            U(i,j)=A(i,j);
        elseif (i==j)
            D(i,j)=A(i,j);
        else
            L(i,j)=A(i,j);
        end
    end
end
%----For Jacobi method----
H=-inv(D)*(L+U);
disp(' The eigen values of the convergence matrix H are ');
E=eig(H)
```

So I will stop here with this one and then I will take the other matrix and this is a 1 2 .3, 2 3.4 5, and this is the right-hand side vector. So let us see what will happen in this case. I will run this one.

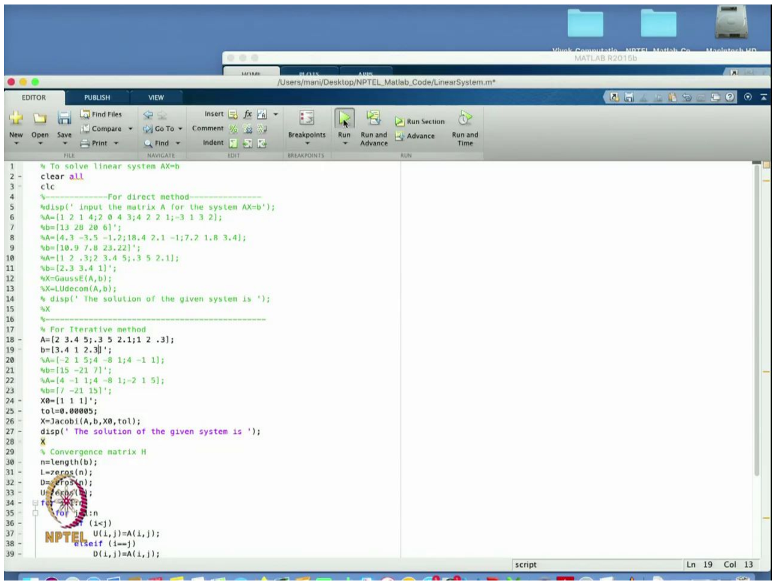
(Refer Slide Time: 17:43)



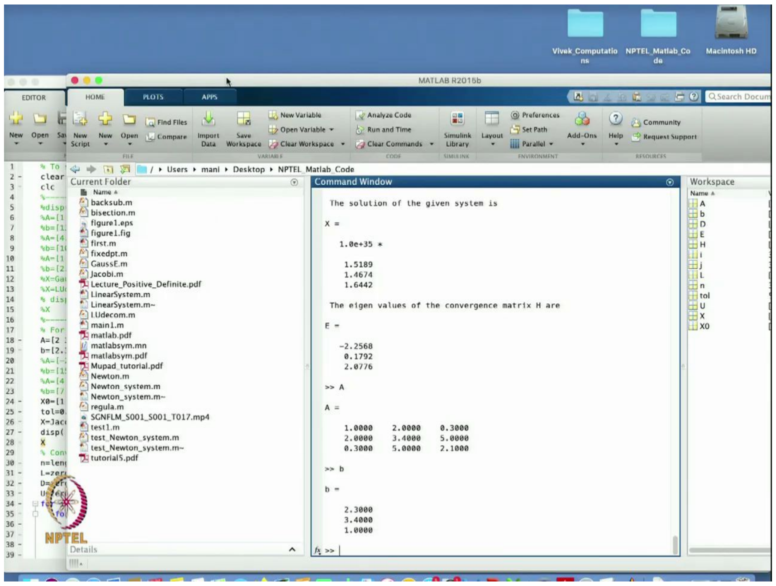
```
err =

   7.5838e+34

 Error at the iteration=99 step

err =

   1.7115e+35

 Error at the iteration=100 step

err =

   3.8625e+35

 The solution of the given system is

X =

   1.0e+35 *

    1.5189
    1.4674
    1.6442

 The eigen values of the convergence matrix H are

E =

   -2.2568
    0.1792
    2.0776

>>
```

So see. Now, in this case also, I am getting my error is very large, infinity. So this system is not going to converge and if you, from here you can see that the eigenvalue is, largest eigenvalue in magnitude is 2.25. So if I see here, then this should be less than 1. If the system will converge then its value should be less than 1. So in this case, my matrix is not a diagonally dominant.

(Refer Slide Time: 18:19)



```
% To solve linear system AX=b
clear all
clc
%-------------For direct method--------------
%disp(' input the matrix A for the system AX=b');
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8 3.4];
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given system is ');
%X
%-------------------------------------------
% For Iterative method
A=[2 3.4 5;.3 5 2.1;1 2 .3];
b=[3.4 1 2.3]';
%A=[-2 1 5;4 -8 1;4 -1 1];
%b=[15 -21 7]';
%A=[4 -1 1;4 -8 1;-2 1 5];
%b=[7 -21 15]';
X0=[1 1 1]';
tol=0.00005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given system is ');
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
    for j=1:n
        if (i<j)
            U(i,j)=A(i,j);
        elseif (i==j)
            D(i,j)=A(i,j);
```

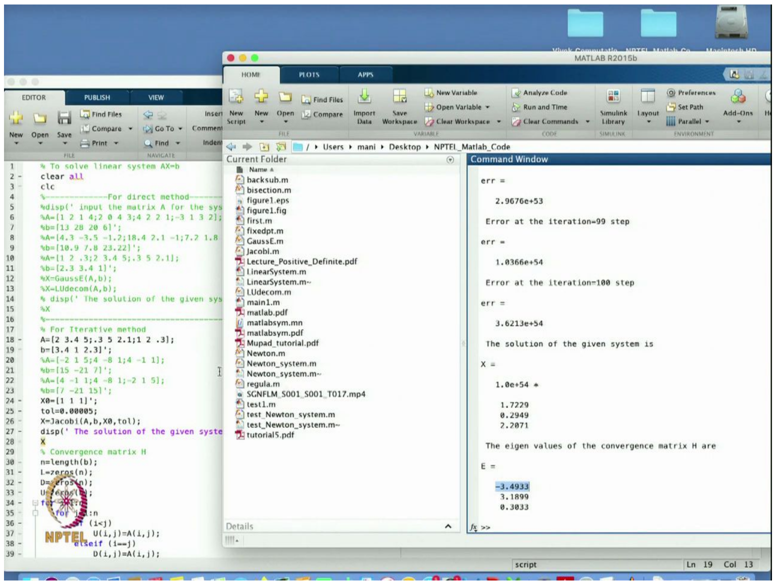


```
 The solution of the given system is

X =

   1.0e+35 *

    1.5189
    1.4674
    1.6442

 The eigen values of the convergence matrix H are

E =

   -2.2568
    0.1792
    2.0776

>> A

A =

    1.0000    2.0000    0.3000
    2.0000    3.4000    5.0000
    0.3000    5.0000    2.1000

>> b

b =

    2.3000
    3.4000
    1.0000

>>
```

Now I will change this one. So let us, if you see from here it is 1 2 0.3, 2 3.4 5, okay, and 0.3 5 2.1. So let us see what will happen if I just interchange the row. So I will take 2 3.4 5, and then I write 0.3 5 2.1 and then I take 1 2 and 0.3. So this I have changed now.

So in the previous, if you see, my A was this one, 1, 2, 3, and my b was this 2.3, 3.4. So now, I have changed this one, now 2 3.4 this, so 3.4 should be here now, so it is 3.4. Then another is 0.3 this, so 1 should be there and 2.3 will be there. So let us see what will happen here because the matrix is not diagonally dominant.

(Refer Slide Time: 19:58)

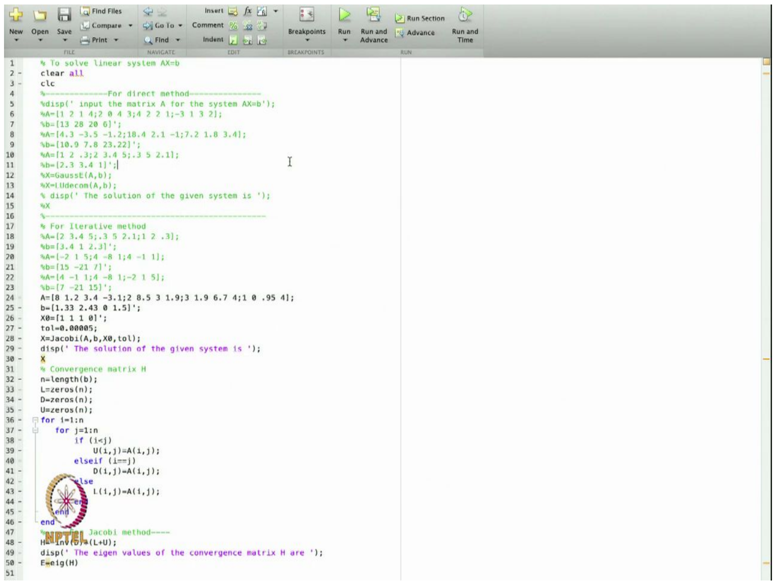


```
% To solve linear system AX=b
clear all
clc
%-------------For direct method-------
%disp(' input the matrix A for the sys
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given sys
%X
%-------------------------------------
% For Iterative method
A=[2 3.4 5;.3 5 2.1;1 2 .3];
b=[3.4 1 2.3]';
%A=[-2 1 5;4 -8 1;4 -1 1];
%b=[15 -21 7]';
%A=[4 -1 1;4 -8 1;-2 1 5];
%b=[7 -21 15]';
X0=[1 1 1]';
tol=0.00005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given syste
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
    for j=1:n
        if (i<j)
            U(i,j)=A(i,j);
        elseif (i==j)
            D(i,j)=A(i,j);
```

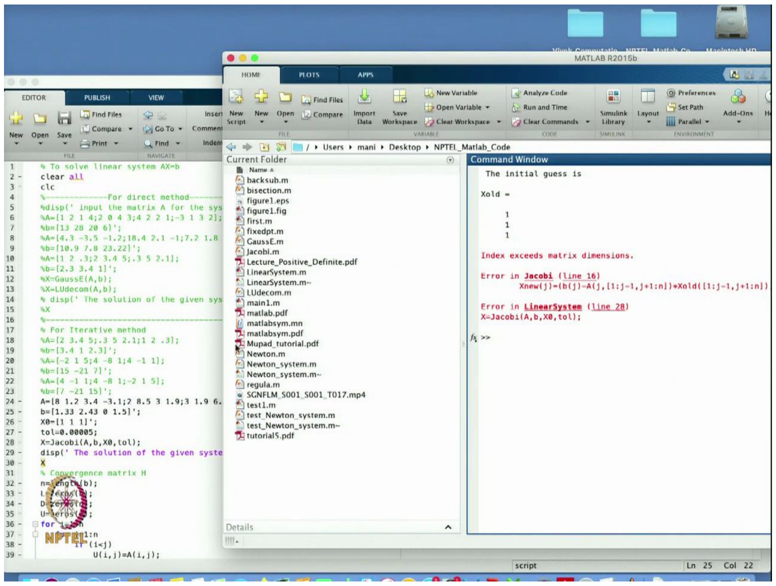
```
err =

   2.9676e+53

 Error at the iteration=99 step

err =

   1.0366e+54

 Error at the iteration=100 step

err =

   3.6213e+54

 The solution of the given system is

X =

   1.0e+54 *

    1.7229
    0.2949
    2.2071

 The eigen values of the convergence matrix H are

E =

   -3.4933
    3.1899
    0.3033

>>
```

So in this case also, you see that the matrix is not diagonally dominant and the Jacobi method is also not converging, and if it is not converging, then necessarily it is necessary condition that you are not getting the eigenvalue, maximum eigenvalue less than 1. So, in this case, I can say that this matrix is not diagonally dominant and my Jacobi method is not converging. So now, maybe I can take another system.

(Refer Slide Time: 20:38)

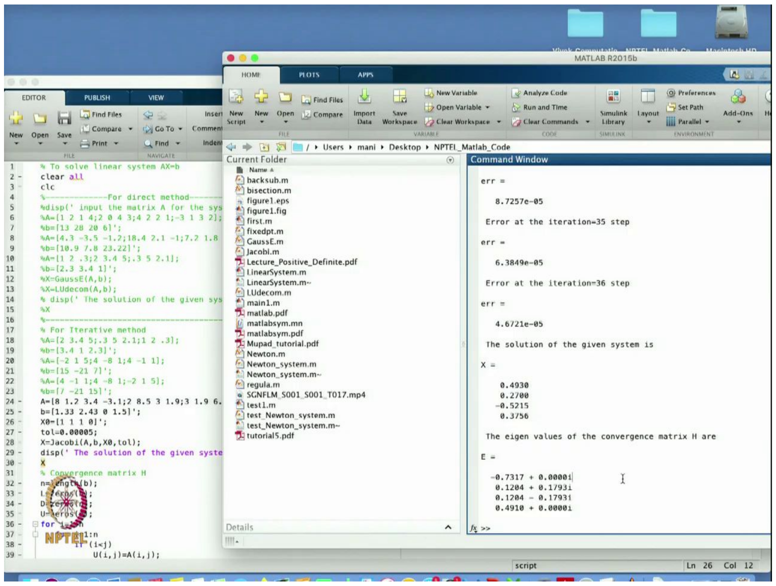


```
% To solve linear system AX=b
clear all
clc
%------------For direct method-------------
%disp(' input the matrix A for the system AX=b');
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8 3.4];
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given system is ');
%X
%-----------------------------------------------
% For Iterative method
%A=[2 3.4 5;.3 5 2.1;1 2 .3];
%b=[3.4 1 2.3]';
%A=[-2 1 5;4 -8 1;4 -1 1];
%b=[15 -21 7]';
%A=[4 -1 1;4 -8 1;-2 1 5];
%b=[7 -21 15]';
A=[8 1.2 3.4 -3.1;2 8.5 3 1.9;3 1.9 6.7 4;1 0 .95 4];
b=[1.33 2.43 0 1.5]';
X0=[1 1 1 0]';
tol=0.00005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given system is ');
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
   for j=1:n
       if (i<j)
           U(i,j)=A(i,j);
       elseif (i==j)
           D(i,j)=A(i,j);
       else
           L(i,j)=A(i,j);
       end
   end
end
%---for Jacobi method----
H=-inv(D)*(L+U);
disp(' The eigen values of the convergence matrix H are ');
E=eig(H)
```



```
% To solve linear system AX=b
clear all
clc
%-------------For direct method--------
%disp(' input the matrix A for the sys
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given sys
%X
%------------------------------------
% For Iterative method
%A=[2 3.4 5;.3 5 2.1;1 2 .3];
%b=[3.4 1 2.3]';
%A=[-2 1 5;4 -8 1;4 -1 1];
%b=[15 -21 7]';
%A=[4 -1 1;4 -8 1;-2 1 5];
%b=[7 -21 15]';
A=[8 1.2 3.4 -3.1;2 8.5 3 1.9;3 1.9 6.
b=[1.33 2.43 0 1.5]';
X0=[1 1 1]';
tol=0.00005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given syste
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
    for j=1:n
        if (i<j)
            U(i,j)=A(i,j);
```

```
 The initial guess is

Xold =

     1
     1
     1

Index exceeds matrix dimensions.

Error in Jacobi (line 16)
        Xnew(j)=(b(j)-A(j,[1:j-1,j+1:n])*Xold([1:j-1,j+1:n])

Error in LinearSystem (line 28)
X=Jacobi(A,b,X0,tol);

fx >>
```



```
% To solve linear system AX=b
clear all
clc
%-------------For direct method--------
%disp(' input the matrix A for the sys
%A=[1 2 1 4;2 0 4 3;4 2 2 1;-3 1 3 2];
%b=[13 28 20 6]';
%A=[4.3 -3.5 -1.2;18.4 2.1 -1;7.2 1.8
%b=[10.9 7.8 23.22]';
%A=[1 2 .3;2 3.4 5;.3 5 2.1];
%b=[2.3 3.4 1]';
%X=GaussE(A,b);
%X=LUdecom(A,b);
% disp(' The solution of the given sys
%X
%------------------------------------
% For Iterative method
%A=[2 3.4 5;.3 5 2.1;1 2 .3];
%b=[3.4 1 2.3]';
%A=[-2 1 5;4 -8 1;4 -1 1];
%b=[15 -21 7]';
%A=[4 -1 1;4 -8 1;-2 1 5];
%b=[7 -21 15]';
A=[8 1.2 3.4 -3.1;2 8.5 3 1.9;3 1.9 6.
b=[1.33 2.43 0 1.5]';
X0=[1 1 1 0]';
tol=0.00005;
X=Jacobi(A,b,X0,tol);
disp(' The solution of the given syste
X
% Convergence matrix H
n=length(b);
L=zeros(n);
D=zeros(n);
U=zeros(n);
for i=1:n
    for j=1:n
        if (i<j)
            U(i,j)=A(i,j);
```

```
err =

   8.7257e-05

 Error at the iteration=35 step

err =

   6.3849e-05

 Error at the iteration=36 step

err =

   4.6721e-05

 The solution of the given system is

X =

    0.4930
    0.2700
   -0.5215
    0.3756

 The eigen values of the convergence matrix H are

E =

  -0.7317 + 0.0000i
   0.1204 + 0.1793i
   0.1204 - 0.1793i
   0.4910 + 0.0000i

fx >>
```

So let us take another system. So I am taking 5 1.2 and 3.4. So this is the first equation I am taking. So let us take the bigger one. I am taking the 7 1.2 3.4 and maybe I am taking -3.1. 3 3, 6; so I am taking 8. So this is the first row I am taking. The second row I will take as 2 8.5 3 1.9. So this is the second row. The third row I take 3 1.9 and then I take 6.7 and then I take 4. So this is another row. And the last row I take 1 0 0.95 and then, maybe I will take 4. So this is the matrix I am taking.

And then I take b vector. So b vector I am taking as, suppose I take 1.33, 1.33 and then 2.43 then 0 then 1.5. So this is a vector I am taking with the column vector. So let us see what will happen in this case.

So this is, the initial guess is because it is 4 4, so I just take the initial guess like this. You can take any initial guess, does not matter. So that is the solution we are getting. So after the 36 steps, you see that my error reduced to this value and that is my solution.

Now, in this case, all the eigenvalues are complex except, no; so we are getting the two eigenvalues real. So it is one eigenvalue this and the other eigenvalue and you can see from there that the largest eigenvalue in the magnitude is this one and that value is coming less than 1.

So because the system is convergent, then it is necessarily that the largest eigenvalue will be less than 1. So from here, I can say that the necessary condition is well satisfied for the Gauss-Jacobi method and my Gauss-Jacobi method is convergent for this matrix.

So if you see that this matrix, I have made is diagonally dominant and that is why the Gauss-Jacobi method has to be convergent because this is the sufficient condition for this one.

So this is all about the code for the Gauss-Jacobi methods and maybe in the next lecture, we will talk about the other code that the Gauss-Seidel method and other related concepts. So thanks for watching this. Thanks very much.