

Scientific Computing Using Matlab
Professor Vivek Aggarwal & Mani Mehra
Department of Mathematics
Indian Institute of Technology, Delhi/DTU
Lecture 15
Matlab Code For Fixed Point Iteration Method

Hello viewers, welcome back to the course. So, today we are going to deal with lecture number 15. So, in the previous lecture we have discussed the Newton-Raphson methods for systems of equations.

(Refer Slide Time: 00:38)

Lecture - 15

Notes Title 02 Mar 20

Complex root of a Equation $F(z) = 0$

$F(z) = F(x+iy) = 0$

for example $F(z) = z^2 = e^z \Rightarrow F(x+iy) = z^2 = (x+iy)^2$

$$= x^2 + (iy)^2 + 2ixy$$

$$(z) = (x^2 - y^2) + i(2xy) = 0$$

$$\Rightarrow \begin{cases} x^2 - y^2 = 0 \\ 2xy = 0 \end{cases} \Rightarrow \begin{cases} f_1(x,y) = 0 \\ g_1(x,y) = 0 \end{cases} \text{ Simultaneous system of eq.}$$

$\Rightarrow \boxed{F(x+iy) = 0}$

$F(z) = F(x+iy) = f_1(x,y) + i g_1(x,y) = 0 \Rightarrow \begin{cases} f_1(x,y) = 0 \\ g_1(x,y) = 0 \end{cases}$

1/1

Now in this lecture we will discuss how we can find the Complex Root of an equation that is my $F(z) = 0$. So, suppose my equation in this one $F(z)$, where I can write my $z = x + iy$. So, my z is a complex number and suppose I have the equation of this type. For example, I have my $F(z) = z^2$ or I have this equal to e^z . So, this type of equation is what I want to solve.

For example, I can write my:

$$F(z) = z^2 = (x + iy)^2 = x^2 - y^2 + i 2xy$$

So, this is the real part and then I can write imaginary part. So, that is my $F(z) = 0$. Now, suppose I take the real and imaginary part.

So, from here, I will get:

$$\begin{aligned}f(x, y) &= x^2 - y^2 \\g(x, y) &= 2xy\end{aligned}$$

So, what is this? It is just a simultaneous system of equations. And in the previous lecture, we discussed how we can find the root of a system equation. So, based on this one, I will find the root of this equation, and I will substitute the value here and then I will get the roots for $F(z)$. So, based on this one, I am able to find my value of $x + iy$. So, I will get the value of x y which is satisfying this relation, I will get the value of xy which is satisfying this relation, and I will substitute back.

So, I will get the value of x and y says that I am getting this equal to 0. So, that is the way we can solve the system of equations. So, and we know from the theory of complex analysis, that any (analytic) function, I can write this function always in the form of $f(x,y)+i g(x,y)$. So, based on this that I can always write a function in this form, then we can put this equal to 0 and then we can solve this equation from here.

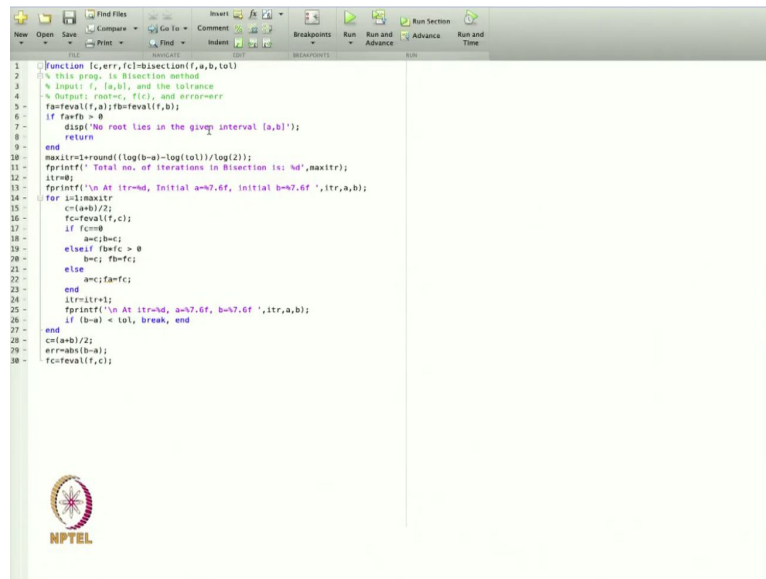
I will get the system of simultaneous equations

$$\begin{aligned}f(x, y) &= x^2 - y^2 = 0 \\g(x, y) &= 2xy = 0\end{aligned}$$

Using the Newton-Raphson methods for systems of equations, we can solve this system and based on this one we can solve this value. So, that is how we can find the Complex Root of the equation $F(z) = 0$. So, this is all about how we can use Newton-Raphson methods for finding the Complex Roots.

Now, let us start doing the code that how we can make the code for whatever the method we discussed in the previous class is. We have just taken the Bisection method; we have coded the bisection method using the MATLAB code. Now, we will make the codes for maybe fixed point methods or Regula-Falsi Method or Secant method or Newton-Raphson method. So, let us write the code for this one and go to the MATLAB command.

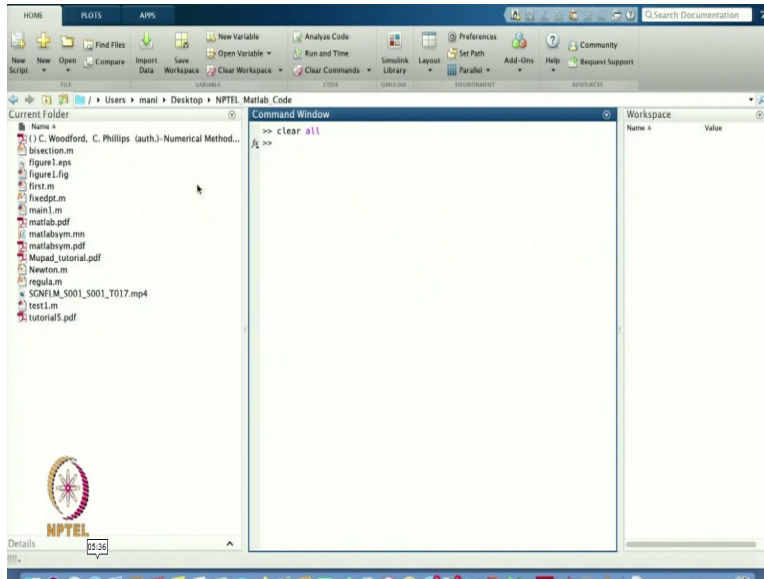
(Refer Slide Time: 05:21)

A screenshot of a MATLAB script editor window. The script defines a function `bisection(f,a,b,tol)` that implements the bisection method. It includes comments in Spanish and English, and uses `fprintf` for output. The script calculates the number of iterations and the error. At the bottom left, there is a logo for NPTEL (National Programme on Technology Enhanced Learning).

```
1 function [c,err,fc]=bisection(f,a,b,tol)
2 % this prog. is Bisection method
3 % Input: f, [a,b], and the tolerance
4 % Output: root=c, f(c), and error=err
5 fa=feval(f,a);fb=feval(f,b);
6 if fa*fb > 0
7     disp('No root lies in the given interval [a,b]');
8     return
9 end
10 maxitr=1+round((log(b-a)-log(tol))/log(2));
11 fprintf(' Total no. of iterations in Bisection is: %d',maxitr);
12 %----
13 fprintf('\n At itr=%d, Initial a=%7.6f, Initial b=%7.6f ',itr,a,b);
14 for i=1:maxitr
15     c=(a+b)/2;
16     fc=feval(f,c);
17     if fc==0
18         a=c;b=c;
19     elseif fb*fc > 0
20         b=c; fb=fc;
21     else
22         a=c; fa=fc;
23     end
24     itr=itr+1;
25     fprintf('\n At itr=%d, a=%7.6f, b=%7.6f ',itr,a,b);
26     if (b-a) < tol, break, end
27 end
28 c=(a+b)/2;
29 err=abs(b-a);
30 fc=feval(f,c);
```

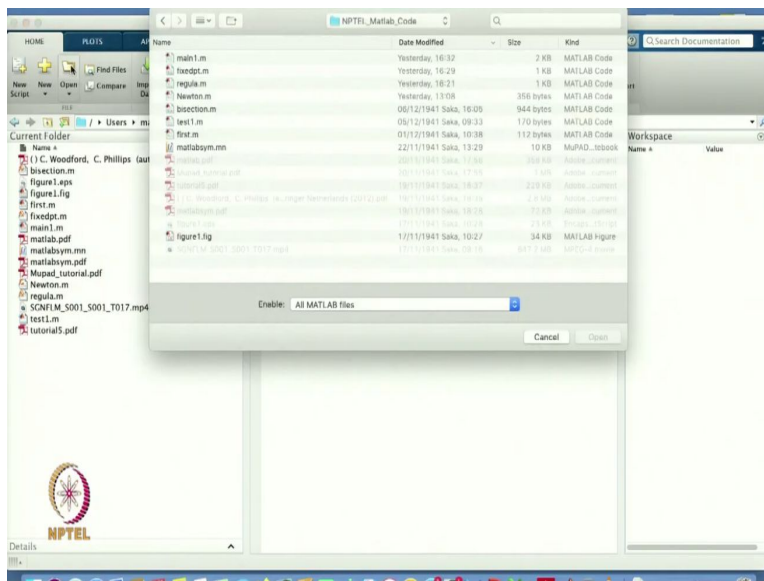
So, if you remember then in the previous lectures, we have discussed the Bisection method and we have called this Bisection method from the file,

(Refer Slide Time: 05:37)



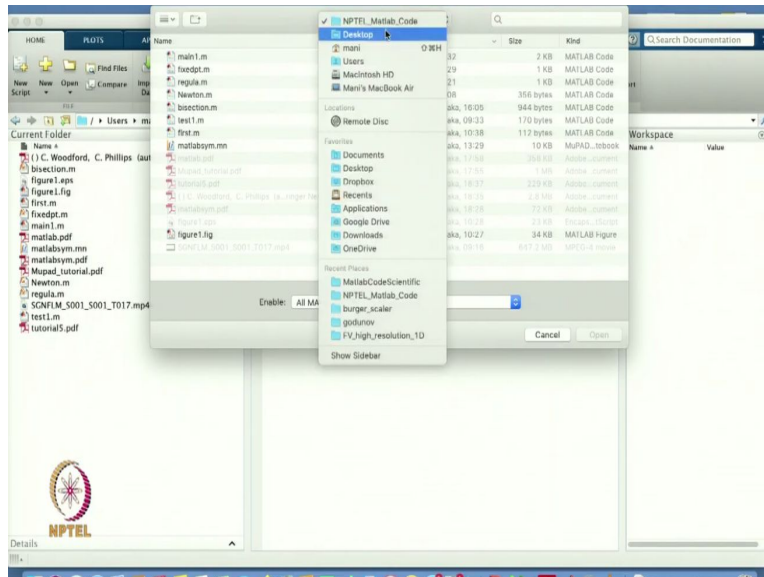
this one and so,

(Refer Slide Time: 05:46)



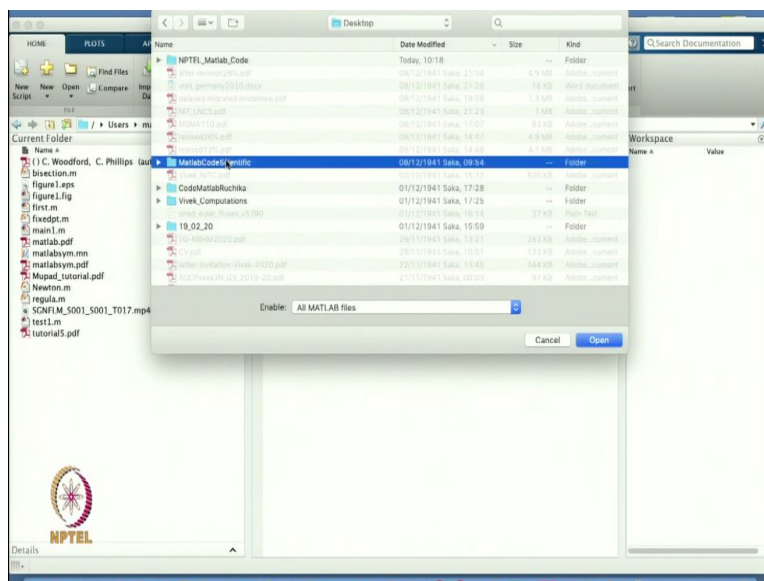
I will go back to this open

(Refer Slide Time: 05:50)



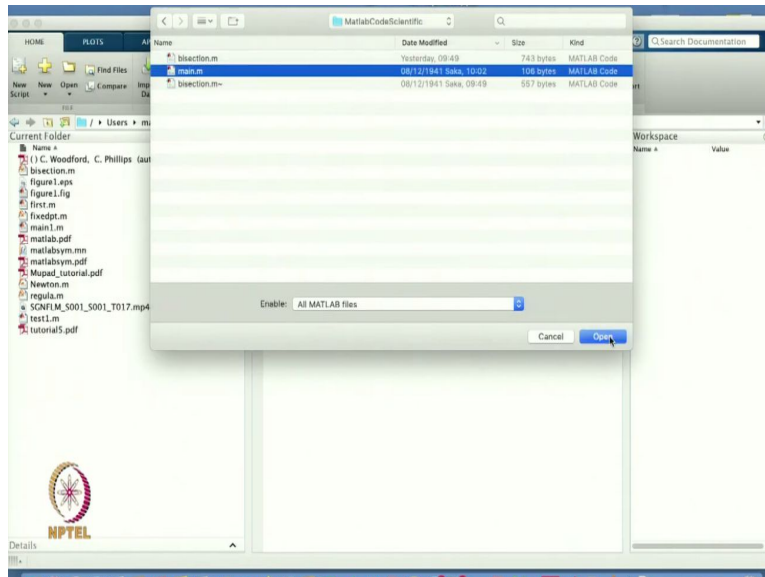
and I will go to the directory which we have created last time.

(Refer Slide Time: 05:54)



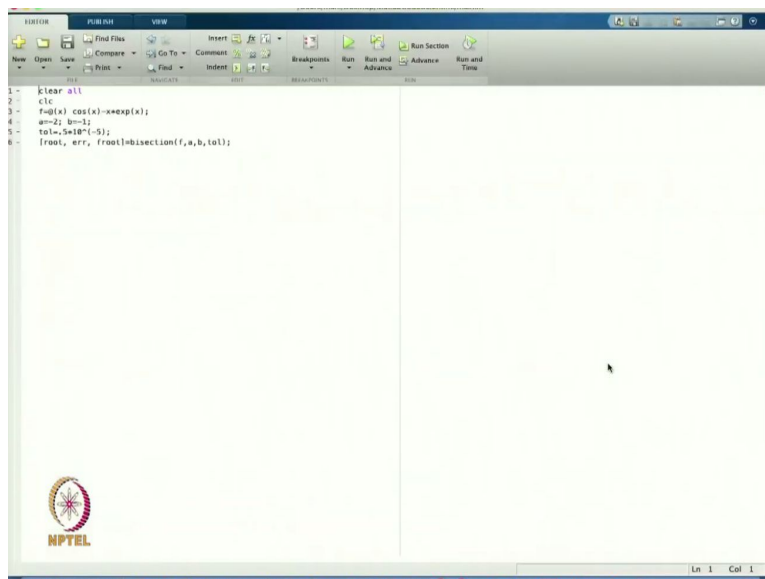
So, this is the MATLAB code for scientific computing.

(Refer Slide Time: 05:56)



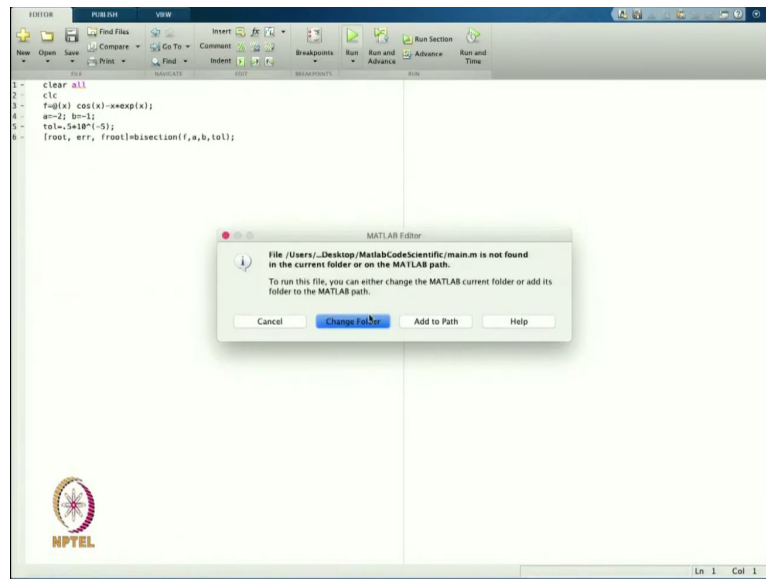
So, in that case we have started this code.

(Refer Slide Time: 06:01)



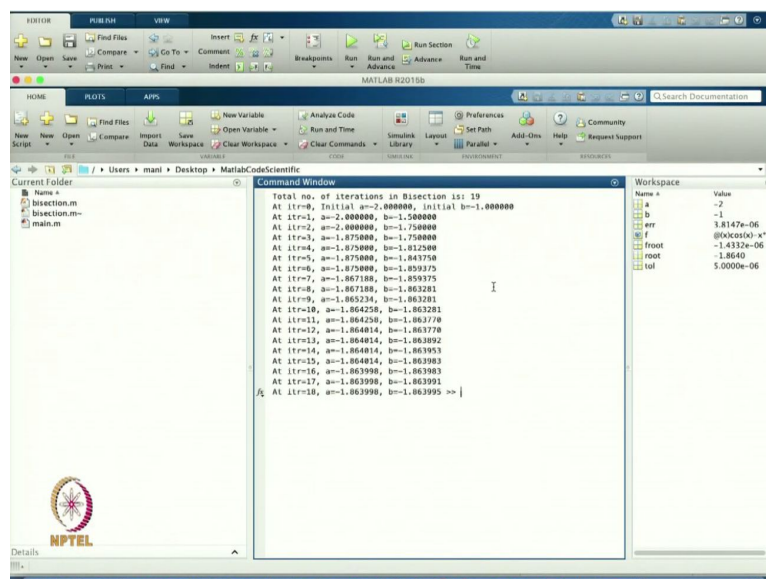
So, now, this is the code we have started with,

(Refer Slide Time: 06:05)



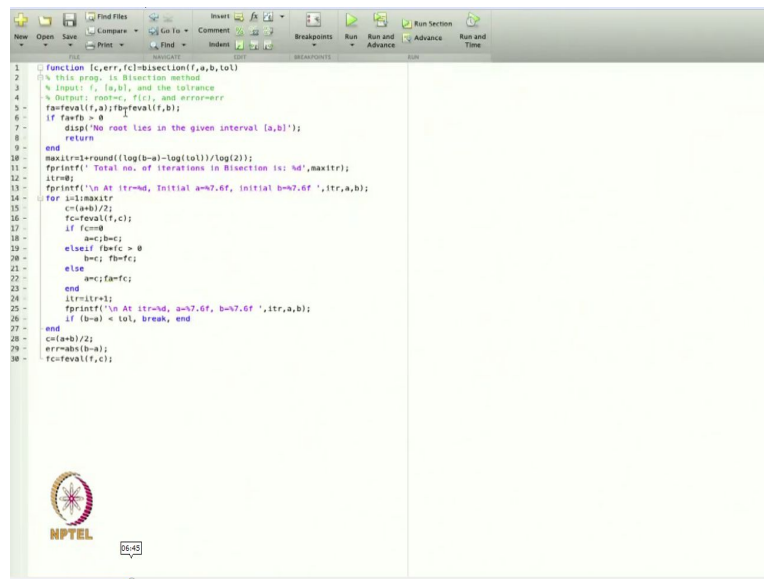
and I suppose I run this code change folder and

(Refer Slide Time: 06:11)



then based on this one I got the solution. So, in the previous lecture, we have started with this one and saw that the bisection method gives you the solution with the 19 iteration. And that is my initial guess $a = -2$ and $b = -1$ and based on this one, I get the value of the negative root and that is -1.86388 . So, that is a way we can find the root for this one.

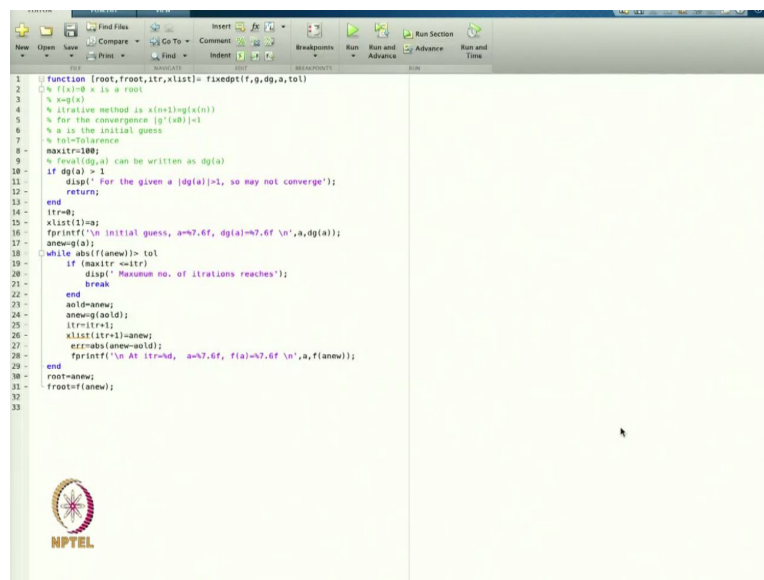
(Refer Slide Time: 06:40)



```
1 function [c,err,fc]=bisection(f,a,b,tol)
2 % this prog. is Bisection method
3 % Input: f, [a,b], and the tolerance
4 % Output: root=c, f(c), and error=err
5 f=feval(f,a);fb=feval(f,b);
6 if fawfb > 0
7     disp('No root lies in the given interval [a,b]');
8     return
9 end
10 maxitr=round((log(b-a)-log(tol))/log(2));
11 fprintf(' Total no. of iterations in Bisection is: %d',maxitr);
12 %end
13 fprintf('\n At itr=%d, Initial a=%7.6f, Initial b=%7.6f ',itr,a,b);
14 for i=1:maxitr
15     c=(a+b)/2;
16     fc=feval(f,c);
17     if fc==0
18         a=c;b=c;
19     elseif fb*fc > 0
20         b=c; fb=fc;
21     else
22         a=c; fa=fc;
23     end
24     itr=itr+1;
25     fprintf('\n At itr=%d, a=%7.6f, b=%7.6f ',itr,a,b);
26     if (b-a) < tol, break, end
27 end
28 c=(a+b)/2;
29 err=abs(b-a);
30 fc=feval(f,c);
```

Now, today we are going to start with another method.

(Refer Slide Time: 06:47)



```
1 function [root,root,itr,xlist]= fixedpt(f,g,dg,a,tol)
2 % f(x)=0 is a root
3 % x=g(x)
4 % iterative method is x(n+1)=g(x(n))
5 % for the convergence |g'(x)|<1
6 % a is the initial guess
7 % tol=tolerance
8 maxitr=100;
9 % feval(dg,a) can be written as dg(a)
10 if dg(a) > 1
11     disp(' For the given a |dg(a)|>1, so may not converge');
12     return;
13 end
14 itr=0;
15 xlist=[];
16 fprintf('\n Initial guess, a=%7.6f, dg(a)=%7.6f \n',a,dg(a));
17 anew=g(a);
18 while abs(f(new))> tol
19     if (maxitr ==itr)
20         disp(' Maximum no. of iterations reaches');
21         break
22     end
23     aold=anew;
24     anew=g(aold);
25     itr=itr+1;
26     xlist=[itr+1;anew];
27     err=abs(anew-aold);
28     fprintf('\n At itr=%d, a=%7.6f, f(a)=%7.6f \n',a,f(anew));
29 end
30 root=anew;
31 root=f(anew);
32
33
```

So, let us start with the new file, this one and today we are going to do the fixed point methods. So, let us write the function and suppose I will today I will get the root. So, I will get the root, then f root, the value of the f (root), then number of iterations and I am introducing the new argument here and that is the x list, x list means that in the previous classes, we have discussed that how we can find the order of the convergence of the method.

So, but to find the order of the convergence of the method I need e_n , e_{n-1} and e_{n+1} . So, three values were the three points I needed. So, I will introduce this one in this lecture. So, that is the output I want and that is equal to, so I will write that fixed point. So, this is the fixed point method I am going to get, so what I need to do, I will pass the f value then I will pass g then I will pass dg, then initial guess and then the tolerance. So what in my f? f is the equation I am going to get the roots g is the function we are going to write as. So, I can write here that fx is equal to 0. So, this is the x is the root, is a root.

```
function [root,froot,itr,xlist]= fixedpt(f,g,dg,a,tol)
```

```
% f(x)=0 x is a root
```

```
% x=g(x)
```

```
% itrative method is  $x(n+1)=g(x(n))$ 
```

```
% for the convergence  $|g'(x_0)|<1$ 
```

```
% a is the initial guess
```

```
% tol=Tolarence
```

```
maxitr=100;
```

```
% feval(dg,a) can be written as dg(a)
```

```
if dg(a) > 1
```

```
    disp(' For the given a  $|dg(a)|>1$ , so may not converge');
```

```
    return;
```

```
end
```

```
itr=0;
```

```
xlist(1)=a;
```

```
fprintf('\n initial guess, a=%7.6f, dg(a)=%7.6f\n',a,dg(a));
```

```

anew=g(a);

while abs(f(anew))> tol

    if (maxitr <=itr)

        disp(' Maxumum no. of itrations reaches');

        break

    end

    aold=anew;

    anew=g(aold);

    itr=itr+1;

    xlist(itr+1)=anew;

    err=abs(anew-aold);

    fprintf('\n At itr=%d, a=%7.6f, f(a)=%7.6f\n',itr,anew,f(anew));

end

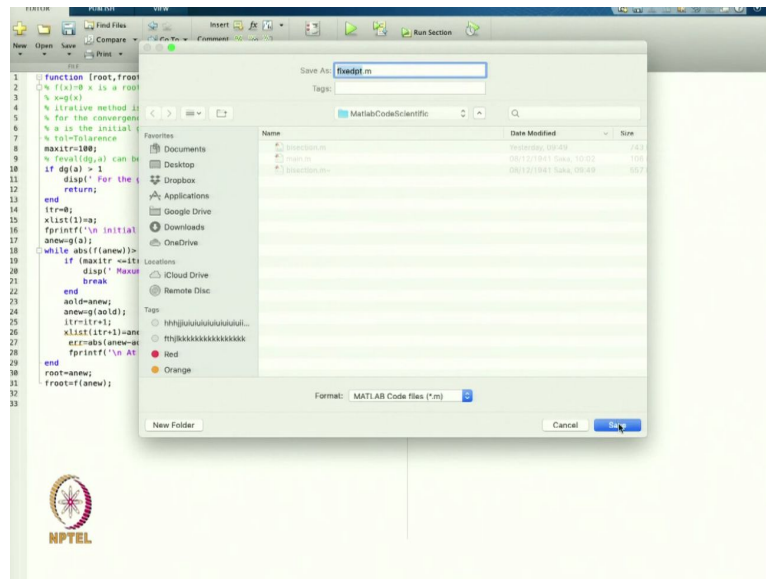
root=anew;

froot=f(anew);

```

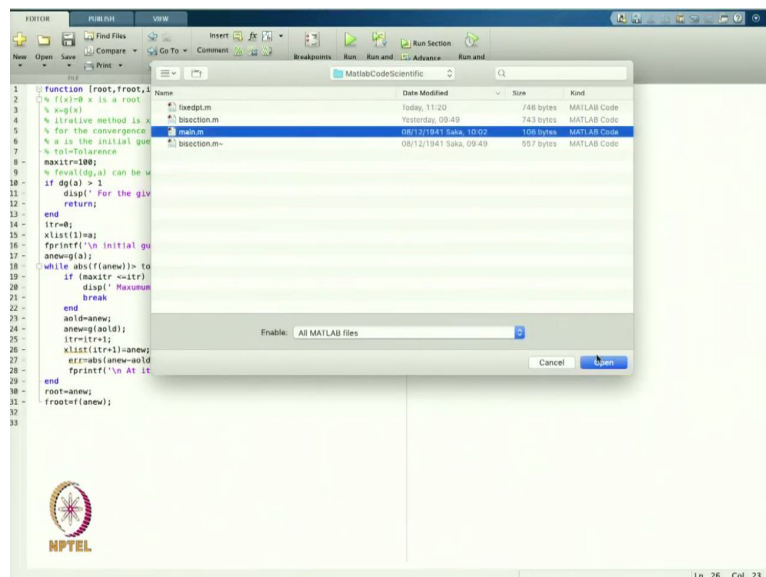
So, it is sure that the maximum number of iterations reached and then it will break the program and then end. So if it comes out from this loop, I will put this as a root of the equation. And this is my I can write as f root so value the function at the root. So, that was the way we can define and from here I can also one thing I also want to put here, x list at itr plus 1. So that is I want to point to a new because I am also starting with 1 from here, so that is why I am putting 1 because here it is starting from 0 it is one. So when it will be 1 it should be 2. So that I am saving here.

(Refer Slide Time: 20:43)



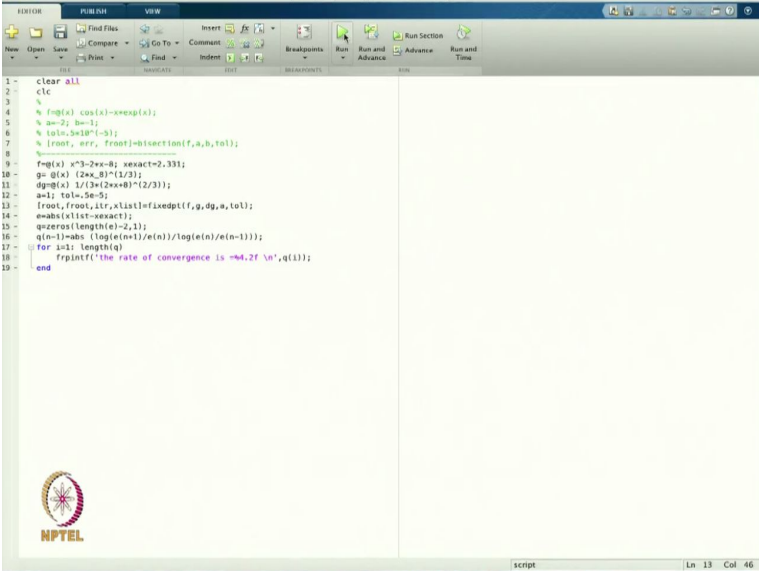
Now I can save this code here, fixed point and then this is saved, now this is saved. So now I want to do the calculation.

(Refer Slide Time: 20:56)



Open, the main program.

(Refer Slide Time: 21:01)



```
1 - clear all
2 - clc
3 - %
4 - % f=g(x) cos(x)-xexp(x);
5 - % a=2; b=1;
6 - % tol=5e-5;
7 - % [root, err, iter]=bisection(f,a,b,tol);
8 - %
9 - f=g(x) x^3-2*x-8; xexact=2.331;
10 - g= @(x) (2*x-8)^(1/3);
11 - d=g(x) 1/(3*(2*x-8)^(2/3));
12 - a=1; tol=5e-5;
13 - [root, iter, xlist]=fixedpt(f,g,dg,a,tol);
14 - e=abs(xlist-xexact);
15 - q=zeros(length(e)-2,1);
16 - q(n-1)=abs (log(e(n-1)/e(n))/log(e(n)/e(n-1)));
17 - for i=1: length(q)
18 -     fprintf('the rate of convergence is %4.2f \n',q(i));
19 - end
```

So let us the open main program and in the main program also we have to do some calculations.

Now, let us start calling the function so I will find my root here, $f(\text{root})$, then itr iterations and xlist. So this is the argument I want from the function, and I am writing fixed point pt. So that was a function I was calling. So error will be e is equal to absolute value I am taking xlist. So this is the xlist minus xexact. So it will give you the value of e the error at each iteration. Now I will get my q, that is the factor I am finding. So my q is equal to I am defining as 0s the length because I need three points always to find out the order of the convergence or the rate of the convergence. So I will define my length of e, so whatever the length is there, minus 2.

So this value I am getting, then I would like f f print f. The rate of convergence is, that is equal to I can write 4.2, maybe f then I can write my new line. And then I can refine this one and then I can define the end. So by this way we can define this function and that is my exact 2. So let us run this code and let us see what is the error here.

```
clear all
```

```
clc
```

```

%-----

% f=@(x) x^3-2*x-8; xexact=2.331;

% g= @(x) (2*x+8)^(1/3);

% dg=@(x) 1/(3*(2*x+8)^(2/3));

% a=1; tol=.5e-5;

[root,froot,itrxlist]=fixedpt(f,g,dg,a,tol);

% e=abs(xlist-xexact);

% q=zeros(length(e)-2,1);

% for n=2:length(e)-1

% q(n-1)=abs(log(e(n+1)/e(n))/log(e(n)/e(n-1)));

% end

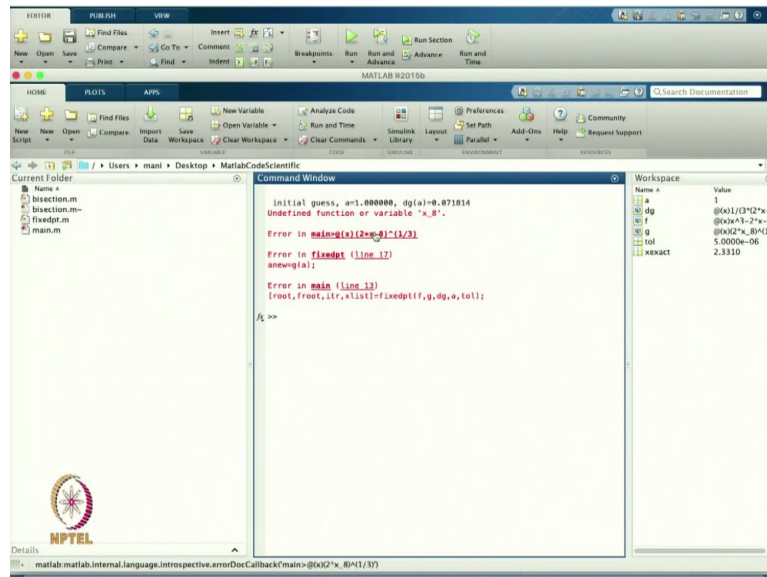
% for i=1: length(q)

%     fprintf('the rate of convergence is =%4.2f \n',q(i));

% end

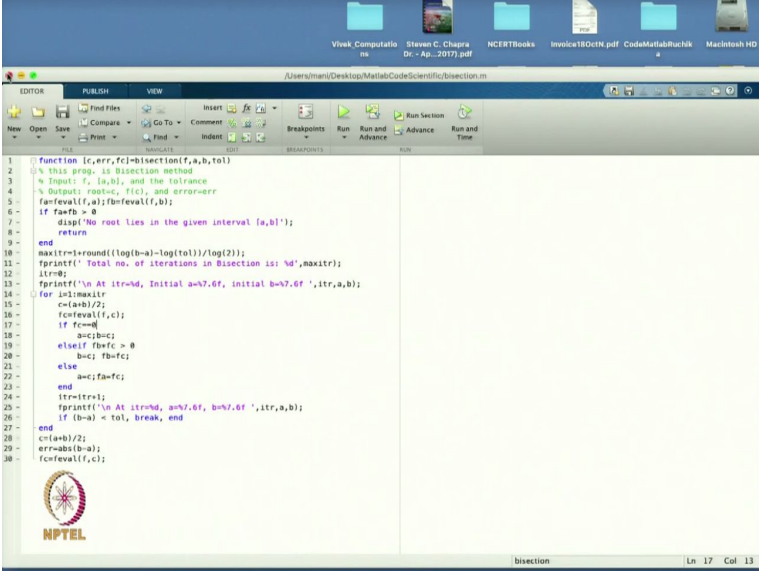
```

Refer Slide Time: 28:24)



So it is showing that the undefined function or variable x eight in the main program, this one okay, okay.

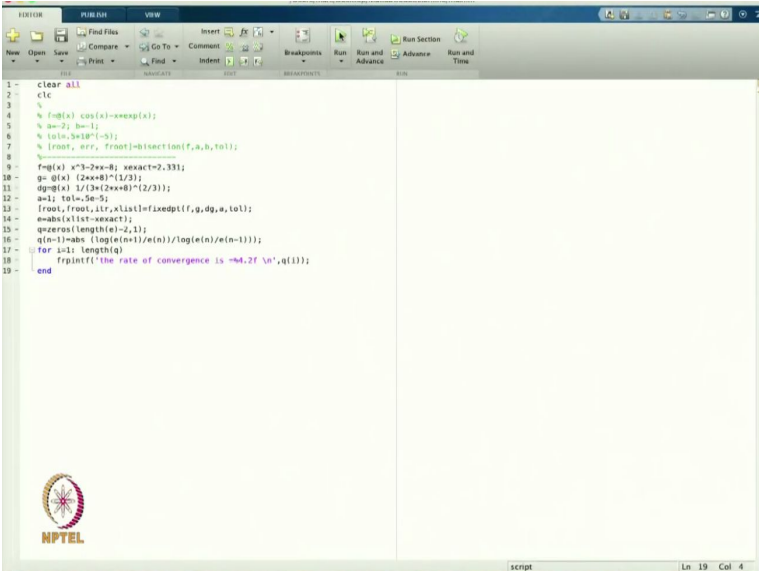
(Refer Slide Time: 28:35)



```
1 function [r,err,fc]=bisection(f,a,b,tol)
2 % this prog. is Bisection method
3 % Input: f, [a,b], and the tolerance
4 % Output: root=r, f(c), and error=err
5 fcn=eval(f,a);fcn=eval(f,b)
6 if fcnfb > 0
7     disp('No root lies in the given interval [a,b]');
8     return
9 end
10 maxitr=round((log(b-a)-log(tol))/log(2));
11 fprintf(' Total no. of iterations in Bisection is: %d',maxitr);
12 itr=0;
13 fprintf('\n At itr=%d, Initial a=%7.6f, initial b=%7.6f ',itr,a,b);
14 for i=1:maxitr
15     c=(a+b)/2;
16     fcn=eval(f,c);
17     if fcn==0
18         a=c;b=c;
19     elseif fcnfc > 0
20         b=c; fcn=fc;
21     else
22         a=c; fcn=fc;
23     end
24     itr=itr+1;
25     fprintf('\n At itr=%d, a=%7.6f, b=%7.6f ',itr,a,b);
26     if (b-a) < tol, break, end
27 end
28 c=(a+b)/2;
29 err=abs(b-a);
30 fcn=eval(f,c);
```

So we should this is a bisection program. So I do not want to keep this program.

(Refer Slide Time: 28:49)



```
1 clear all
2 clc
3 %
4 % f=@(x) cos(x)-exp(x);
5 % a=-2; b=1;
6 % tol=3e-5; %
7 % [root, err, froot]=bisection(f,a,b,tol);
8 %
9 fcn(x)=3-2*x-8; xexact=2.331;
10 g=@(x) (2*x+8)^(1/3);
11 dg=@(x) 1/(3*(2*x+8)^(2/3));
12 a=1; tol=5e-5;
13 [root, froot, itr, xList]=fixedp(f,g,dg,a,tol);
14 e=abs(xList-xexact);
15 zeros=length(e)-2,1;
16 q(n-1)=abs (log(e(n+1)/e(n))/log(e(n)/e(n-1)));
17 for i=1: length(q)
18     fprintf('the rate of convergence is %4.2f \n',q(i));
19 end
```

this is my main program. So that is the function and this is to $2x + 8$, so that is the function we have defined $2x + 8$, so $2x + 8$ that the function we have defined so let us see now again there is some error.

(Refer Slide Time: 29:23)

```

MATLAB R2015b

Command Window

Initial guess, a=1.000000, dg(a)=0.071814
At itr=1, a=0.308941, f(a)=
At itr=1, a=0.038315, f(a)=
At itr=1, a=0.004700, f(a)=
At itr=1, a=0.000770, f(a)=
At itr=1, a=0.000071, f(a)=
At itr=1, a=0.000001, f(a)=Undefined function or variable 'n'.
Error in main (line 18)
q(n-1)=abs(log(f(n))/e(n))/log(f(n)/e(n-1));

f1 >>

Workspace

Name      Value
a         1
dg         0.071814
f1         1.1110e-0221
f10        0.0000e+00
f100       -1.0681e-06
f1000      0.0000e+00
f10000     2.3307
f100000    5.0000e-06
f1000000   2.3310
f10000000  [1.2 3089.2 3281]
  
```

Now it is showing undefined function or variable n Okay, so here we will see from here initial guess is a is equal to 1 and the value of d at dg the derivative g at 1 is 0.01. So in that case, okay, we will going to get the root of the equation, okay? So that is the value of a the root we are heading toward.

(Refer Slide Time: 29:45)

```

MATLAB R2015b

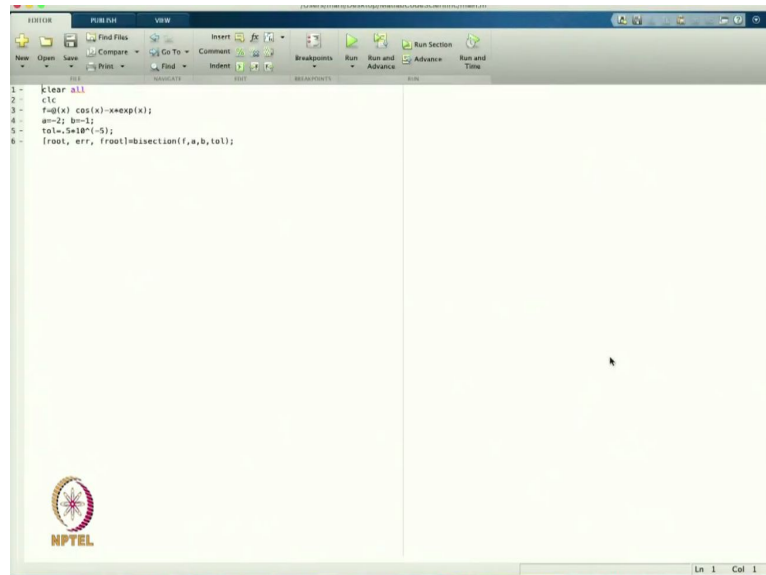
Command Window

function [root,rootn,itr,xlist]= fixedpt(f,dg,a,tol)
% f(x)=0 is a root
% xug(x)
% iterative method is x(n+1)=g(x(n))
% for the convergence |g'(x)|<1
% a is the initial guess
% tol=tolerance
% maxitr=1000
% feval(dg,a) can be written as dg(a)
if dg(a) > 1
    disp('For the given a |dg(a)|>1, so may not converge');
    return;
end
itr=0;
xlist=[];
fprintf('\n Initial guess, a=7.6f, dg(a)=7.6f \n',a,dg(a));
anew=a;
while abs(f(anew))> tol
    if (maxitr-itr)>0
        disp('Maximum no. of iterations reaches');
        break
    end
    anold=anew;
    anew=anew-dg(anew)/dg(anew);
    itr=itr+1;
    xlist=[itr+1;anew];
    fprintf('\n At itr=%d, a=7.6f, f(a)=7.6f \n',itr,a,f(anew));
end
root=anew;
rootn=f(anew);

fixedpt
Ln 31 Col 15
  
```


So this is my fixed point method we have defined. So let us see why it is not giving the value of dg , at iteration. So I have to put itr here, then the value of a and then the value of f a new and this one. So that is the error here, I have saved this one.

(Refer Slide Time: 30:19)

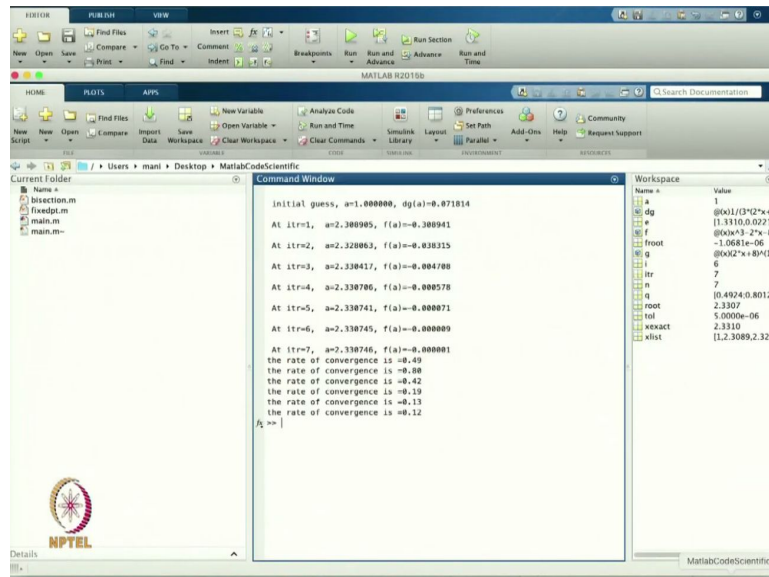


```
1 - clear all
2 - clc
3 - fup(x) = cos(x) - x*exp(x);
4 - a=2; b=1;
5 - tol=5e10*(-5);
6 - [root, err, froot] = bisection(f, a, b, tol);
```

Now we will write from here for n minus 2, so this is length of e minus 1. So this is the value we have taken because this is the value of the length. So, I will write my for loop and e starting from 2 to length of e minus 1 and that is end, now let us see.

(Slide Time: 31:06 to 31:33 Speaker Voice not Available)

(Refer Slide Time: 31:37)



So, now it is running and that is the final solution. So, from here we can see that I will start with my initial guess that is one and at that place that value the $dg < 1$. So, it is going to converge so, then I will get my value of a this value of fa is this. So, with iteration after the same iteration I will get the value of a that is 2.3307 and the value of the function so this is the value of the function, I am not putting the absolute value.

So it is 0.00001. So, it is up to tolerance. So it is becoming less than the tolerance and the rate of convergence, I started with the initial guess. So, it was showing 0.49, then it was 0.48, 0.42, 0.19 then 0.13, then 0.12. So, in this case my rate of convergence is always coming less than 1 because it cannot be greater than 1. So, this is the way we found the rate of convergence.

Because we know that in the Fixed Point Methods, our rate of convergence is always unique means the linear the 1. So, it may happen that sometime we will get the rate of convergence depending on the value of that what is the value of ga. So, based on the value of these ga we can find the rate of convergence.

So in this case, my rate of convergence is 0.12. So maybe in the next class, we will go for the other methods. So this is a Fixed Point Method. In the next class, we will go for the Regula-Falsi and the Newton Raphson method. So thanks for viewing, thanks very much.