Scientific Computing Using Python

Professor. Vivek Aggarwal and Professor. Mani Mehra

Department of Mathematics

Indian Institute of Technology, Delhi

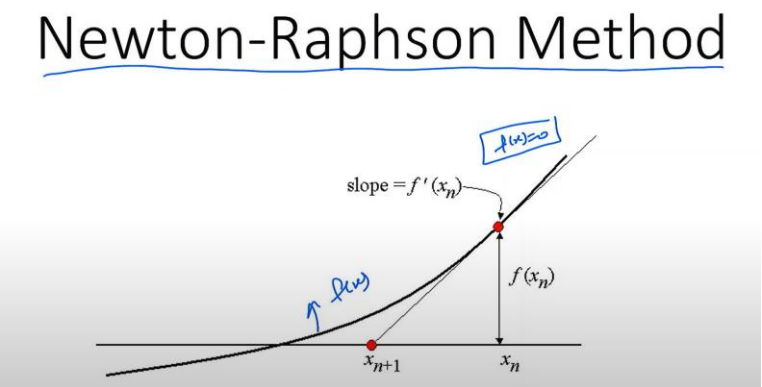
Lecture No. 11

Welcome to Scientific Computing Using Python. So, as in the previous lecture, we discussed the Secant method and the Regula Falsi method to find out the roots of a function. So, today we will discuss another method, which is Newton's Raphson method. So, let's start.

So, Newton's Raphson method is a very famous method because its order of convergence is very fast. So, let's see why it is fast in Newton's Raphson method. As the figure shows, till now we have done many methods, and in them we have seen that, like Secant method and Regula Falsi method, in that we did not use the derivative of the function. We used its finite difference, which is approximation is used . But what is happening in Newton Raphson method , we are using tangent. So, look, in this, Newton's Raphson method is the same thing—that it is to find the approximate root of the non linear equation but what is there in this is that it is based on the derivative of the function, which was not there till now, to iteratively improve the estimate of the root. And Newton's method is to use the tangent line at a concurrent approximation to find out the next approximation. So, its iterative process is defined like this with the help of this one.

So now the main question is how to start this, because here we can take zero, one,two , three and in this way we will take values and we will get different iterations. So, the question is, how did this formula come about? So, as I have shown in this figure, what is happening in Newton's Raphson method—like this is a function, like if there is a function a f(x) equals zero, then we will have an equation, then we have to find its root. So, this function, f(x) is a function, which is not a equation, this means we have to find out this equation and this value of function is given.

(Refer slide time: 3:05)

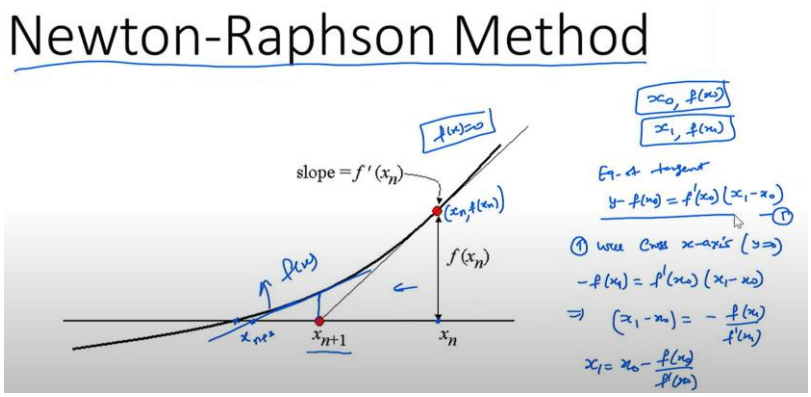


So, what do we do? I took an approximate initialization, I named it xn. What did I do? I saw the value of function at xn, so the values are this. The value which will be its coordinates will be xn and f(xn), these become its coordinates. So, as soon as its coordinates were formed, what did we do? We drew a tangent on it. So, look, we are drawing a tangent. As soon as we draw this tangent, we will see where it intersects the x-axis. So, look at this point, the tangent is intersecting at this point. So, the next point which comes, we will take the next approximation. You will see that earlier here was an approximation, and now it has moved here. Now we will see at this point. What will I do? I will take the tangent here. This tangent comes—where will it intersect? So, this point, next approximation has come—I will call it x n+2, right? Then take it here, and in this way, we will get the roots.

So now the main thing is that we took the initial approximation. So, look at that. We took an initial approximation. I took it as x0 . So, our point which came is (x0, f(x0)). Next point of ours is given, we took a derivative in this, and what will we get after that? We will get x1, and we will get f(x1). So, what are we doing? I am writing the equation of the line. The equation of tangent—what will we write as y minus y1, which means f(x1) is equal to f dash. We have to find f dash. See, we need the derivative of the old one, so we will say f at x0, and here I have x1 minus x0. So, we know that we can find the equation of the tangent line. Now this tangent will cross the x-axis. So, this tangent is—I will name it one. So, one will cross the x-axis, because where it crosses the x-axis, there we get the next approximation. What we will do there?. Put y as 0. So, if I put y=0 in equation one. This will become minus f(x1) equal to f dash x0, x1 minus x0.

So, from here, we have to see what our x1 is now. So, from here, the value that we have is—I can write x1 like this here. So, see what happens from here. We have x1 minus x0—I can write this as f at x1, f dash x1. Okay, so from here we get the x1 that comes to us: x0 minus f(x) by f dash x0. This is our—the values that we have are these given points. Sorry, this point is zero. Okay, this point is zero, so if this is the next y, then the equation line is y.

(Refer slide time: 7:06)



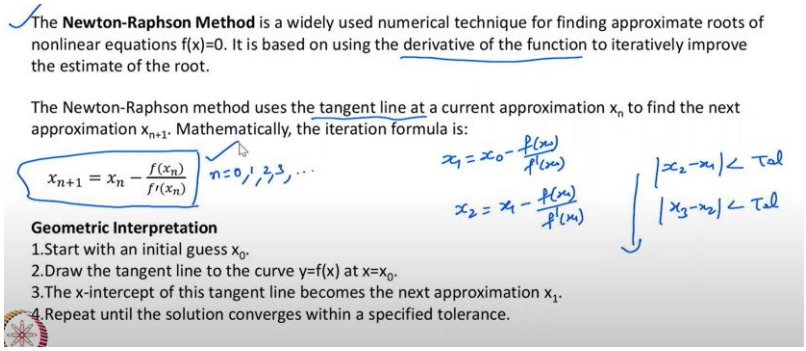
So, what are we doing? We have a point—we have (x0,f(x0)) . Okay, let's write the value at that point. I want to write the value. I don't know what equation it is, so I will arrange it on y—we don't know. So, there is a point and its derivative. So, what equation do we write in it? Let's write it like this: y minus this is equal to f dash x0, x1 minus x0. So, I will remove the x1 as well and write x minus x0. Okay, so now this has come, and from here we have the values x minus x0. So, from here we have x1. So, here we have this iteration. Now I wrote it as x1.

Okay, so now what we have—now see, we had x0. So, what did I do? X0 minus f(x0) divided by f dash x0. So, the values that we have, f dash x0, that have come, take the derivative of the function here.

Now, if we show it like this, then it comes. So, you will see what we had in our formula. Here, if I put A as zero, then our x1 will come equal to x0 minus f(x0) divided by f dash x0—this will come. Then, if we see x2, then this will come: x1 minus f(x1) by f dash x1. So, like this, we will keep taking the values, and our iteration will keep on repeating.

Now, how do we have to stop this? So, what will we do? I will see whether the modulus of x2-x1 is less than the tolerance or not. If it is not, then x3 will come out x3 - x2. We will see whether it is less than the tolerance or not. So, like this, we will keep checking, and its iterations will go on. Okay, so this formula tells us that in this, we are evolving the derivative.

(Refer slide time: 9:27)



But if you see, in this—like there was bisection method, which was bisection method, regula falsi, secant method—so in all of them, we needed two initial guesses to start. But in this, you see that on our right, there is only xn on the side, which means that we need only one guess to start with. So, if we see, in the fixed point method, we also need an initial guess. In this also, we need an illiterate guess—only one is needed. It can have the drawback in that there should be a derivative involved.

So, what is the geometric interpretation? Start with the initial guess. We need an initial guess, xn. Draw the tangent line to the curve at this point. The intercept of this tangent line becomes the next approximation, that is called x1, and repeat until the solution converges with the specified tolerance. We have seen that tolerance is a very important thing, and generally we define .5 into 10 raise to the power some n. Okay, n, so this is our number of digits, up to where we need correction. So, n is a significant digit—we need correction up to that point. In this, we need n-digit correction.

So, what are the steps in this? The main thing is: choose the initial value, compute it, and check the convergence. So that xn minus x1, or the value of f(xn), can be seen that even if there is less tolerance, we can stop, because as this xn approaches the root, the value of the root will be exactly zero. But we are not getting the exact root—we are getting some value, right? So, if we are getting the value, then the value of this will be very small, very small. So, what do we say? If it is a tolerance, then we should consider it to stop, or if it is greater than the tolerance, then it should go on.

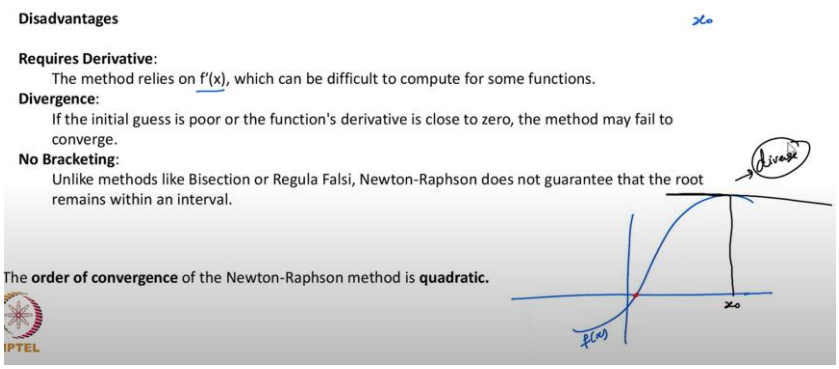
Now, is there a fast advantage in this? The method converges quadratically near the root, meaning the number of correct digits is approximately double with the iteration. And we have seen the formula—it is very simple, it is computationally efficient, and the derivative is easily calculated. And if the derivative is being calculated easily, then after that, we will say that it converges very quickly.

It will converge. The disadvantage towards roots is that the derivative is this. So, the method relays on the derivative, which can be difficult to compute for some functions. We did not take such a function that it would be very complicated, and divergence also happens if the initial guess is poor. But if we chose an initial guess x0 and I chose it, then it turned out that this method did not converge at all.

For example, if we have a method, function is something like this—okay, this function, which is f(x) . Now, what do I have to do? I have to find its root. So, its root is this, and this is—I will it with red—this is its root. Now, what we have to do is that we need the initial guess. So, what did I do? I needed the initial guess, so what did I do? I took it here. I took it at this point, and here I took the derivative. I saw the tangent. Where will we find this tangent? Where will it cut the x-axis? Not anywhere. And if it cuts on this side, then it will cut after going very far.

So, in this case, what will happen is that whatever happens will diverge.

(Refer slide time: 14:01)



So, the solution that will be there will not be found. We will get divergence. So, what will happen in this case is that our iteration will never converge. So, we have to keep this thing in mind—that where we have to take the initial guess and how to take it.

No bracketing—unlike methods like bisection, Rega Falsi, Newton Raphson—then there is no guarantee that the root remains within the interval. So, there is no guarantee of that, because it does not satisfy the intermediate value theorem. That means it is not a requirement.

Now, what is the benefit of this? In this case, the order of convergence of the Newton method is quadratic. So, what does quadratic mean? That the error at a given n+1 step is equal to some constant c en power 2. So, this will come to us. So, our method keeps quadratic convergence. So, this thing will give us quadratic convergence.

Now, let us see how it is giving quadratic convergence. If I want to prove it a little bit, if I want to do a derivation—that how the quadratic convergence is coming, like we did in the fixed point—after that we saw that the convergence in its fixed point was linear. So, if we

want to see it the same way in this, then we know that this is our method. So, the scheme that we have is this.

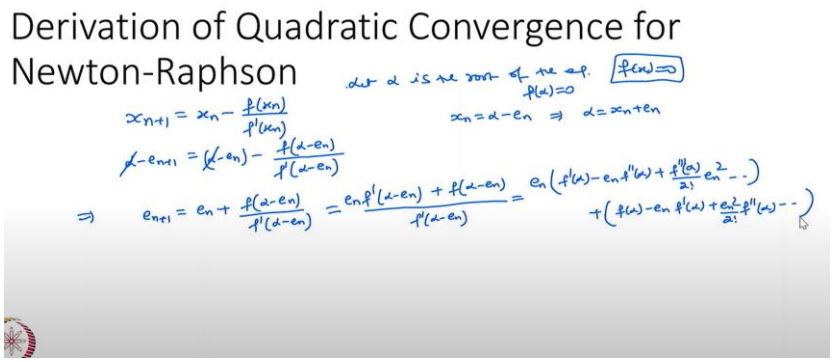
So, like we have this: some x n+1 is equal to, it xn minus f(xn), f dash xn—this is taken. Now, let alpha be the root of the function f(x)=0. The root of this equation that we have is the root of the function or the root of the equation. Let me write it down. What does it mean? That f of alpha will be zero. So, we know that if we do iteration xn, then xn if alpha will be minus the some error—only then it will converge. What does it mean? That our alpha is approximation plus some error.

So, if I do the same thing here, then my—this will become alpha minus e n+1, error at the n+1 step, and this becomes our alpha minus en, en meaning error, f dash alpha minus en. So, this equation that we have will become—so alpha, alpha gets cancelled from here. So, if we see, we have the error at the n+1 step is equal to error at the n step plus f alpha minus en divided by f dash alpha minus en. We can also take it like this: f dash alpha minus en, into en which it was, plus f alpha minus en divided by f dash alpha minus en.

Now see, this function is continuously even differentiable, because only then we are taking the derivative. So, we assume that this function is continuously differentiable—all its derivatives are done. So, if it is so, then we can expand this function with the help of Taylor series. So, we can write its Taylor series. I wrote it with Taylor series, so f dash will come as f dash alpha minus en, f dash double dash alpha plus—now we have f double dash triple dash alpha by 2 factorial en square. Okay?

So, it will go on like this: plus f alpha minus en, f dash alpha plus en square by 2 factorial, f double dash alpha. This is how it comes to us. So, if we see, what have I done?

(Refer slide time: 19:18)



I just expanded it, okay, divided by the same derivative. The same derivative will be f dash alpha minus en, f double dash alpha plus en square by 2 factorial, f triple dash alpha. This is how it goes. Now, there is also en here, okay. Now, we know that alpha—this value has become zero because we have got the root—so this will be equal to zero.

One of our en is also coming out, and one of the en is here, okay. So, if we look at it carefully, this will cancel out from this and this. Now, we have the square of en coming and the cube of en coming here. So, what do we say? Let's ignore all the powers of as a cube, okay—higher—because if we square, then all the squares are coming. So, what do we have to do? Higher. But before that, we also need to see what to do with this value below. So, if we calculate this as well—

Now see how I am writing this: this will be our remaining value. So, minus e to the power n, it will be square will become f double dash alpha, e to the power n cube. So, I will, plus it from here. I have en square by 2 factorial, f double dash alpha. Next, I will ignore this as well. And here, f dash alpha minus en, f double dash alpha plus en square by 2 factorial, f triple dash alpha. If I take it up, it will become power minus 1.

So now, see—if we expand all of this, then minus 1 will come. So, I will take f dash alpha value. So, if we calculate this carefully, you will see that in the end, it will become e, e to the power n of a square by 2 factorial f double dash alpha divided by f dash alpha. So, actually, if we could have taken f dash alpha here, then minus 1 would have gone up. So, from here, I have this value. We will take this value because f dash alpha is common from here.

After that, this f double dash alpha by f dash alpha will come. I will expand it with a binomial. Then we will get this. So, if I do it like this, then I got these values. So, from here, we can see that the values that we had e n+1 have become now, minus f dash alpha, f double dash alpha by f dash alpha by 2 en square—so we have this, right? So, we have this value. So, what will happen as soon as this value comes?

We can write from here that our iteration has become e n+1 is equal to some constant c, e n square, where c is minus half f double dash alpha over f dash alpha. So, this is what we have, okay. So, from here, we get to know that the order of convergence of Newton Raphson—that it has come to know that 2, two means that it will converge very fast, okay.

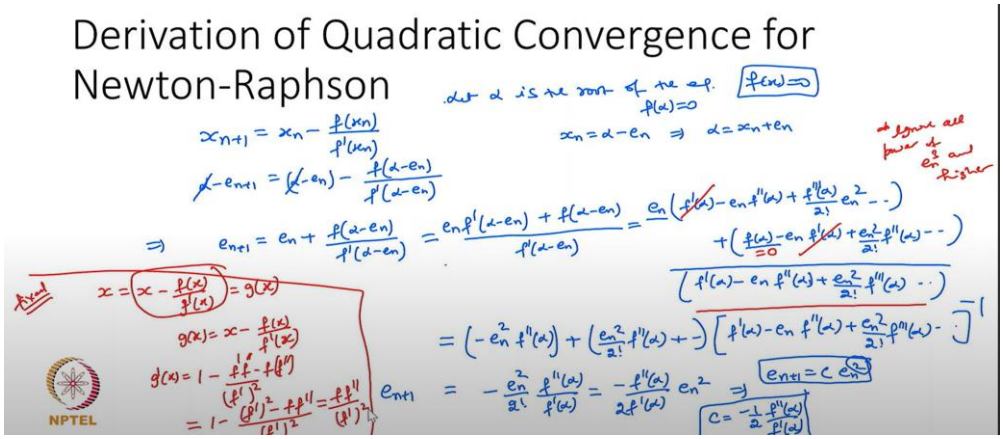
So, in this, the only problem that we had in the second property was that it may not converge, even if the initial approximation that we have is okay. If it is not there, then it turns out that it did not converge at all. So, what can we have in this case?

Now, what can I do? I will take its help. Let's see, our scheme was Newton Raphson , x is equal to x minus f(x), f dash x. So, what do we do? If you remember, I would assume it to be g(x) from a fixed point. So, if we have to do it from a fixed point as well, then what will we do? We will assume g(x) to be this.

So, g(x)—if this is there—then I can write g(x) is x minus f(x) by f dash x. Now, we know that if we take g dash x, then what will happen? One minus f dash square will come, f dash, f minus f double dash whole square—this we have, okay. f dash square, f dash f, minus f , f dash f will come double— sorry— will not become square, it will become double derivative. It will become derivative twice—this.

So now, we do this, and from here, this comes. f dash dash—so we have this— one minus the f dash square minus f f double dash over f dash, the whole square. This comes taking the derivative. So, you see, if I expand it a little more, then what will be left with us? If I take this f dash up, it will get cancelled. So, ultimately, the last thing that will remain is f f double dash divided by f dash whole square.

(Refer slide time: 26:44)



So, if we want that our convergence should be sure, then what will we do? g dash x becomes our f f double dash by f dash whole square. So, let's take the modulus value—that it should be less than one. So, if this is less than one, we have seen earlier that at a fixed point, it definitely converges. So, if we take this condition for the function, then on the basis of this, we will choose the initial value. Then we will come to know that this Newton method of ours is definitely going to converge.

So, whenever we take the x_0, we will substitute the second derivative, first derivative inside it, and put such values and see whether its value is less than one or not. If it is less than one, then we will choose it. Then we will come to know that this will converge.So, this is the way we can find out how Newton's Raphson method works. So, we have found the roots with this.

Now, what we have done with Newton's Raphson method till now is for an equation. We had an equation: f(x) = 0, and we found its roots. So, the Newton's Raphson method is that we can very easily extend it to a system of nonlinear equations. So, Newton's method can be extended to solve systems of nonlinear equations which are very easily, involving two variables. So, what we did was, I extended it to two variables. For example, if we have a system question, there is an f which is a function of two variables, and g is a function—that too of two variables. So, we have these two variables and two equations.

So, if we want to solve this, it means we need a root whose values—if I take its root, then the root will be of the form: some α, β. Then α—we will put α instead of x and β instead of y, and both will be equal to zero. So, we will say that this is its root.

So, what we do is solve it in this. Now, as if we see in the previous one, what was there in this was that the derivative was known. So, the first thing we have to do—the second thing we have to do for the first time—is that we have found the derivative for a function or an equation. Now, we have two equations, so we have a system equation. So how will we find its derivative?

Because if we have to apply Newton's Raphson method, then we need this, and so if we have to make an iterative method, then we need how to make an iterative method. So, what we do is copy this and see how it will work on this system of nonlinear equations or how it will be extended.

So now, see what we have. We have a variable—I will write it with capital X. It is a vector. It has two variables, two components: x and y. Because there are two variables, it has a function. I took F(X). What would that be? In this case, that too would be 2-dimensional. So, its first component would be this, and the second component would be this.

So, this is our function. So, it means that this is our system of non-linear equations. I can write it like this: F(X) = 0. So, see, just like we were writing it here, we wrote it down. So, this is our equation, okay, in the vector form. This is our equation in the vector form. So, now I have to solve it to find the roots.

So now, what do I do? Look, I will do it with Newton's Raphsonn. So, in the same way, I will try to write Newton's equation, i.e., xn+1 is equal to x n minus f(xn) divided by f dash xn. So, looking at this, if I try to write something similar to this, then look: what will be its iterative process? Its iterative process is becoming: X n+1, capital X n minus. Now, we need the derivative, okay? So, we know that this is a function. Now, this F is a function of two variables, x and y. And our functions f and g are both functions of two variables. So, if we have to find the derivative of these, then we have to do one thing, and that is the Jacobian. Okay, in multivariable calculus, if we take the derivative, then we have to find the Jacobian. So, we represent it as J . In this case, let's see what is Jacobian.

Look, now what we have is the function f, so del f by del x with respect to one variable, del f by del y of the other variable. So, this is of one, and the other is our g. So, del g by del x and del g by del y. So here we have the derivatives of this, because we will have four derivatives.So, now we have the derivatives. So, if we have to find out the derivative for the vector equation in this way, then we have to take the help of the Jacobian. So, it means we have to take Jacobian.

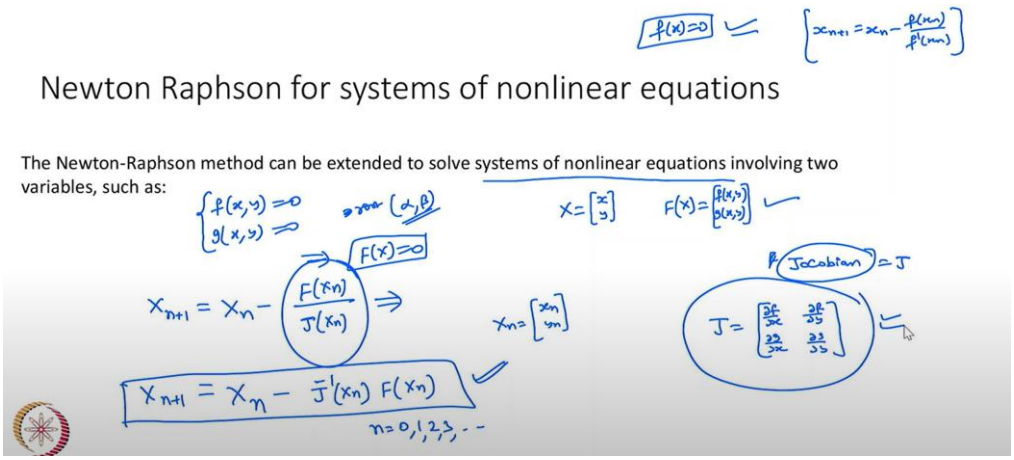
So, what am I doing? In this Jacobian is a matrix. So, I can either write it like this: F(Xn) divided by jacobian, okay? Let's take J(Xn). We have to take the points, but this is a vector. I have divided it by a matrix, so this is not the correct notation. So, I will change this notation, okay?

So, I will write it like this: now it becomes Xn minus J inverse, I write because it is a matrix, so the inverse will be a 2x2 matrix, and our vector is 2x1, so it will get multiplied easily, okay? So, in this case, what we will have is that we will get a vector Xn., we have a vector, okay, so Xn and X n+1 will come to us next.

So in this—what would happen in this—what does Xn mean? A vector xn. So n is 0,1,2,3 and so on. So, this is our notation. So, this is our Newton Raphson for system of nonlinear equations.

In this, we can easily find out. All we have to do is find the Jacobians. Alright, we will find out their Jacobians, and after that we will solve it.

(Refer slide time: 35:07)



So, now these things that we have done—the Jacobians—so let's do one: how can we find the Jacobians? So, let's take an example.

Alright, so now suppose I take a system of equations. So, I took a function:

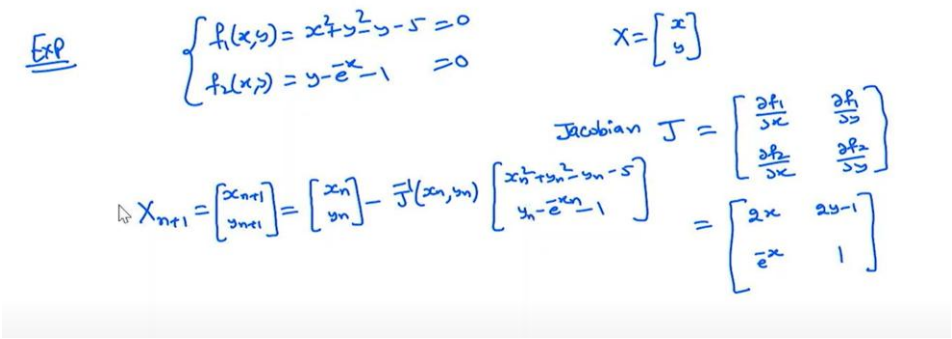
$f_1(x, y) = x^2 + y^2 - y - 5$
$f_2(x, y) = y - e^{(-x)}-1$

Now, what we have to do is find their roots, alright? Which satisfy both the equations. This is the condition we have. So, what does it mean? If I put this equal to zero, then we have this system of nonlinear equations formed. Now I have to find its roots.

So, we have done a lot of system of linear equations before. We used to make equation matrix form and solve it, but we have a nonlinear one, so how do we solve it? Okay, so for this, what do we have to do now? We have to solve it by using the iterative method. So, this is formed. We have function one and function two, and we have two equations. So, the vector X that we have is x and y. And now, what do we have to do is, the first thing that we have to do is that in this case, find the Jacobian. So, in this case, the Jacobian will be, the J what? What is the Jacobian? We had written del f1 by del x, del f1 by del y, del f2 by del x, and del f2 by del y. This is our Jacobian matrix. So, we will calculate it. Del f1 by del x means taking the partial derivative of it, so we will have 2 x. Now, we have to take the partial derivative of it with respect to y. Okay, so this will come to us 2y-1. Now, we have come here. Now, we have to take its partial derivative with respect to x, so the exponential will come. So, this will become plus. So, this will become e minus x. Okay, and we have to take it with respect to y, so this one. So, this is our Jacobian.

So now, how do we have to solve this? To solve this, see what we have to do now. Capital X n+1, this means x n+1, y n+1 is equal to capital Xn. So, capital X n means this minus Jacobian inverse and on what we have to take xn, yn? Because in this, both xn and yn are coming into F. So, F is ours: xn square plus yn square minus yn minus 5, and the power of yn minus e to the power minus xn minus one. This is what we have. Okay, so this will become a 2 by 2, and this is 2 cross 1. This is column. There are two rows. So, if we calculate from here, then 2 cross 1 we will have a vector. It will be minus from this, so what we have is a new value.

(Refer slide time: 39:24)

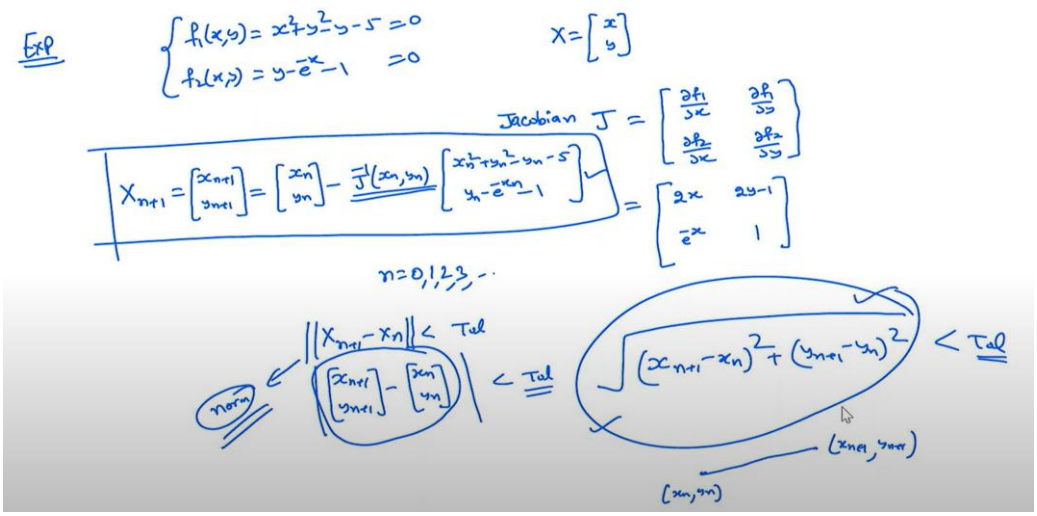


So, you saw that from here we can calculate it from only what we have is work left is , this has to deal with: the Jacobian , vectors and finding the inverse of the Jacobian. So, our iteration will be zero, one, two and so on. Now, you can see what I have to do at each iteration. I have to find the inverse of a matrix, which is a two by two matrix, and multiply it, and do matrix multiplication, and then minus it. So, this work will have to be done at each step. Only then will we be able to solve it.

Now, what is happening in this? What do I have to do? That tolerance means how to stop. So, the stop will happen only when we have X n+1 minus Xn. The difference between them is less than the tolerance. Now, what is this Xn, is a vector. Now, this has become a vector. We have x n+1 and y n+1 becomes a vector minus xn, yn also becomes a vector. The difference between these two has to be found. So, we know that if we have two vectors, then how can we find the distance between them? So, we will go ahead and discuss a little about the norms: vector norms, matrix norms. So, in this, we have to do matrix norms. Do not apply it, because these are vectors, so we will apply vector norms. So, we will see. We will manage with the euclidean norm. What will we do? x n+1 minus xn whole square plus y n+1 minus yn whole square, these are 2 values, okay? They must be less than tolerance. So, this we have Euclidean norm. Ee can take other norms as well: one norm, infinity norm. All right, so we can calculate all these. So, we are taking this. So, when this value gets reduced by the tolerance, then it will stop with us.

So, this thing, if you see it later, then we will make it like this. So, what it becomes, these norms become—we call it—meaning length. So, what is the distance between these two? If we have to find the distance between x n+1 and x n, then we will have to define it in norms. So, we minus and define the norm. So, from here, we know what it is doing. There is a value here: x n, y n, and there is a value here: x n+1, y n+1. There are two points in a plane. Okay, so if there are two points in the plane, then if we have to find the distance between them, then we have been using this method till today.

(Refer slide time: 42:28)

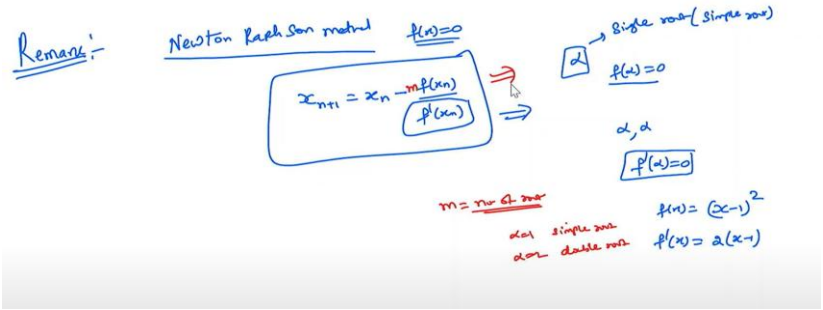


So, we will use this method, and by this, we will get to know how much distance is left between the two. So, we will keep on reducing this distance, and our convergence will continue. So, we can apply this Jacobian method to solve the system of equations. You can do it for two, then for three. In that, Jacobian will become 3 by 3. If we do it for four, it will become 4 by 4. So, it will become quite complicated. So, we can solve this very easily.

One more thing that I want to say is the remark. What is in the remarks is that if we see that the Newton-Raphson method that I am using for f(x)=0, in that, we saw that this method is x n+1 is equal to x n. I am using it for a simple for one equation, and it becomes this, okay. So, what did we do? In this, we have a root alpha. So, we know that f of alpha will be zero, but this is possible only if it is a single root. So, it is a single root, or we say simple root.

But what if it is a repeated root? If we find an equation, the roots that come out are alpha, then another alpha comes, so it becomes a repeated root or a double root. So, in that case, you saw that f dash alpha also becomes zero. Okay, like we have a function polynomial, f(x), (x-1) to the power square. So, it has two roots: one and one repeated roots. Okay, so if we take its f dash derivative, what does it become? 2 x minus 1. So, see, it also has one root. It also has 1 root, and it also has 1 root. So, if this is happening, we have a repeated root. So, what happens in that case is that as we approach alpha, its value will become very small. The approximation almost goes to zero. So, what happens , the method we applied its convergence slows down. So, the method will give very slow convergence. It will be applied. Okay, so if we want that its convergence remains the second order, okay, if we want to form the second order, then what do we do? We put an a here, m, which is the number of roots which we have repeated. For example, if the root alpha is simple, then alpha equal to 1, for simple root, alpha = 2 for double root, alpha = 3 for triple root. So, we put the value of m. So, right now we were finding for the simple root, so we were writing one here. So, we mean automatically one was here. But if we have a double root of it, and if we are solving it with Newton's method, and we want that its convergence should be large, second order, then for that, we will have to put m here. So, we will have to keep this thing in mind: that if there is a repeated root, then our convergence will be affected. So, all these things which we have done with Newton's Raphson.

(Refer slide time: 46:30)



So now, we will discuss its Python code, that how can we do this with Python. Now, I start its Python code. So, this is Newton method. We have opened it. So, what did we do? We are doing the same thing in all the programs that we have made till now. Okay.

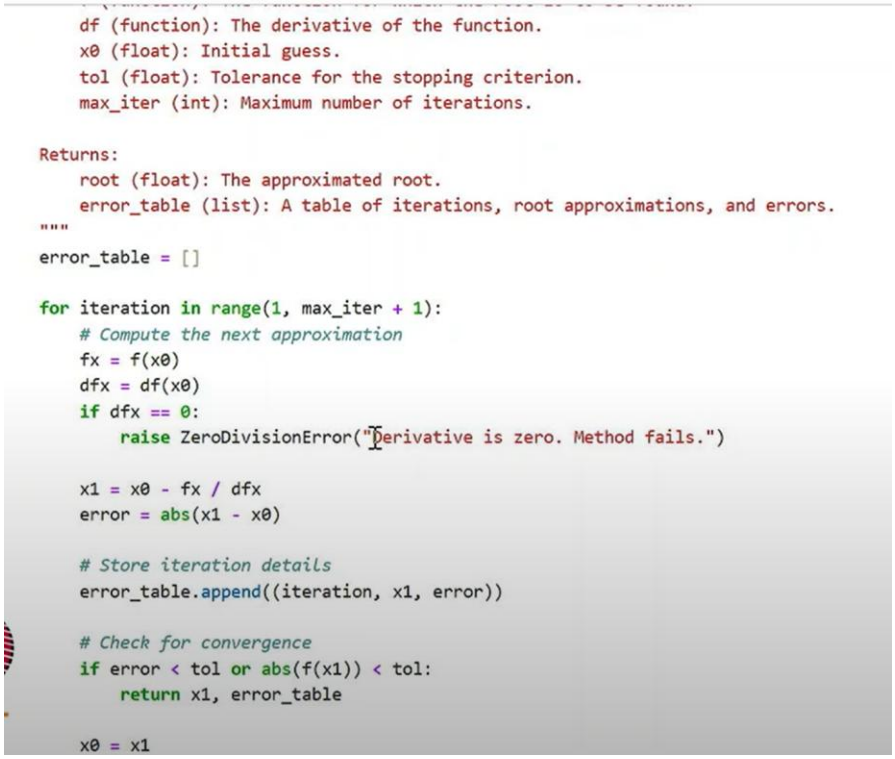
(Refer slide time: 47:01)

```
import numpy as np
import matplotlib.pyplot as plt

def newton_raphson(f, df, x0, tol=1e-6, max_iter=100):
    """
    Implements the Newton-Raphson method to find the root of f(x) = 0.

    Parameters:
        f (function): The function for which the root is to be found.
        df (function): The derivative of the function.
        x0 (float): Initial guess.
        tol (float): Tolerance for the stopping criterion.
        max_iter (int): Maximum number of iterations.

    Returns:
        root (float): The approximated root.
        error_table (list): A table of iterations, root approximations, and errors.
    """
    error_table = []

    for iteration in range(1, max_iter + 1):
        # Compute the next approximation
        fx = f(x0)
        dfx = df(x0)
        if dfx == 0:
            raise ZeroDivisionError("Derivative is zero. Method fails.")
```

I just made it 0.5 into 10 to the power minus 6. This iteration is done. Now see, in this, we need only one initial data. So, the initial x0 has come, the tolerance has come, the maximum iteration has come. What will we get in the return? We will get the root, and we will get the error in the table.

So, what did we do? We applied the same method that we used. f(x), which is the value of f at x0. We are writing dfx as the value of derivative f at x0. So, if the value of the derivative is zero, then what will we do? We will write the derivative here. A message will come that the derivative is zero. Method fails.

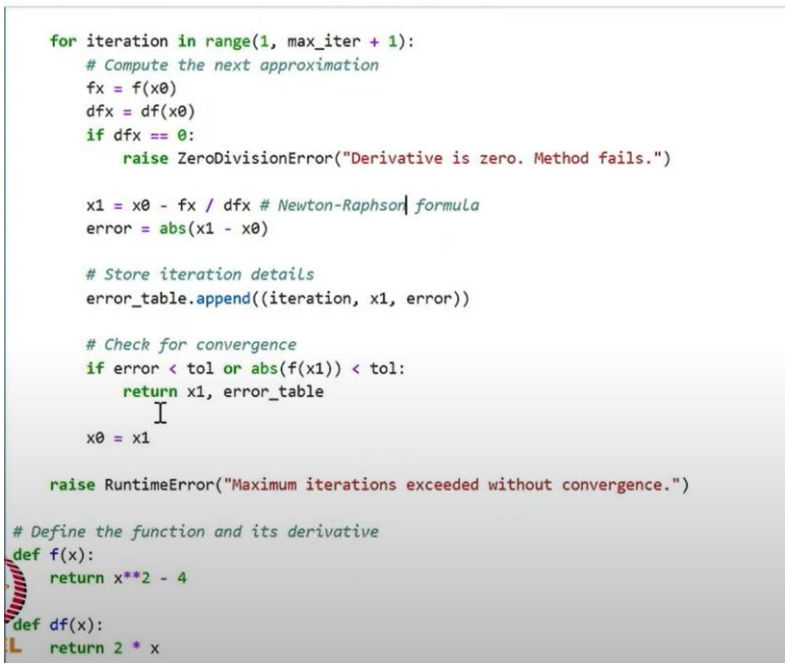
(Refer slide time: 47:44)



Okay, so now this method will fail, and we will not be able to do anything. So, what will we do? We will repeat it and change it. Now, whatever we applied here, we have taken it out. I can write it as Newton formula, Newton Raphson formula.

This formula was applied. x1 equal to x0 minus f(x) and divide by dfx. And the error came in both of these. Just like this, we did it. Stored the iteration by appending iteration in iteration first column, iteration second column, we have the root. See the error in third.

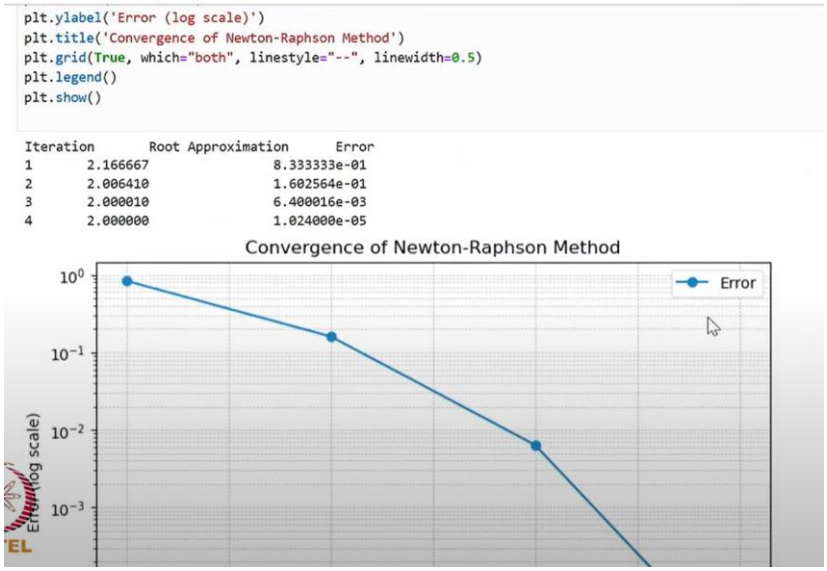
(Refer slide time: 48:34)



And we keep checking that where convergence is less than this. Only then do it, whenever it is less than tolerance, then stop it. So, x0 is equal to x1. So, like this, it will keep repeating for us. Okay, like in this, error less tolerance or absolute value is less than tolerance. Then it returns x1 and error table. Otherwise, what will it do? As long as it is large, it will keep running.

So, we are calling the function from here. So, define the function and its derivative. So, I defined the function here. So, it comes f(x) is equal to return x square minus 4. Okay, and like this, we took the derivative. So, I took the derivative and calculated its values. Now, I want to see its root. So, its root. So, we know that its root is 2. Okay, but I give the initial guess 3 and run it and see what is happening. So, as soon as I ran it, we have this.

(Refer slide time: 49:54)

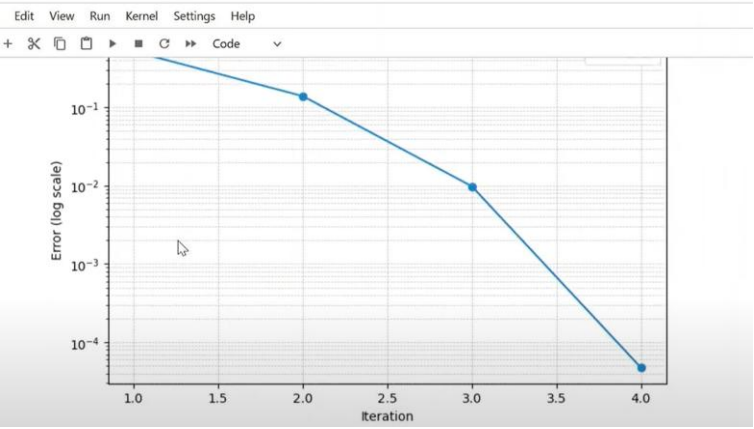


So, its convergence came to this, and that's fine. And in even four iterations, we got this solution.

Now see, if it goes up to even two digits, then the second comes in the same way that we had. So what did I do? I took the initial data three. If I take, suppose, four and then check that even four digits are fine, so it has reached this and its value came. So you will see in this that the convergence that we have, and how fast the convergence is happening. We were doing the same thing with regula falsi and secant. So in the secant, we saw that the 2 was coming on the seventh iteration. Okay, but what is happening in this is that fast convergence is coming. Now suppose don't take this and take this. I define it here. The only problem in this is that we have to take derivatives as well, so that has to be taken care of. Now I am writing. I have done the control V. So we have this x cube minus 2x minus 8. Okay, so the value that we have is x cube - 2x - 8. If I write it here, then I get x cube - 2x - 8. So we have a function, so I have to write its derivative as well, and the derivative 3x square - 2. So when we did this above, we had defined this root of this. It was coming out to be 2.3 something. So we also have to write 3x square - 2 there. Okay, so now the function that I have defined here, return this, so let me comment, what will happen here, the return will be 3 into x square - 2. Okay, so 3x square - 2, we have its derivative.

Now we have to see what the initial value is. So let me take the initial value three and run it. Let's see, we started with three and I got the answer in four iterations.

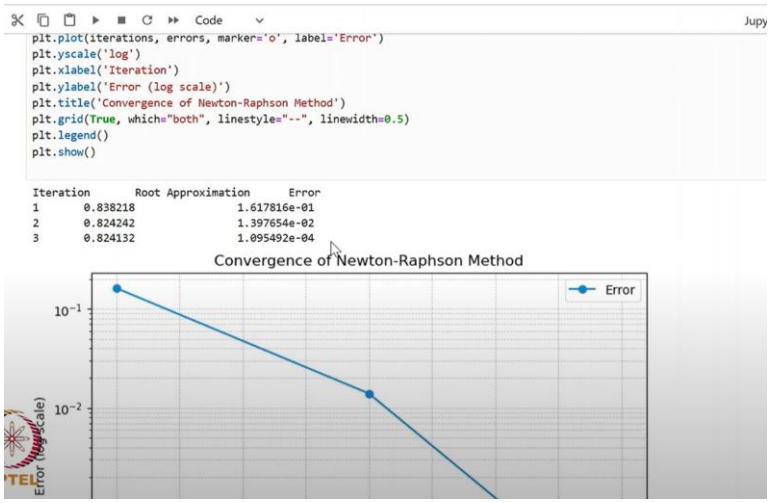
(Refer slide time: 53:04)



This is happening because of Newton Raphson. If I take the initial value, suppose one, see, we took nine iterations and from there we have the solution has arrived. So see how fast it is converging. It is converging from here, and so from here you will come to know that the line that we have for linear, we will get a line below that. Like we saw in the bisection, what was happening in the bisection, the method was going from here, so what will we say in that case? That its convergence is less than one. In this, it is going from above, so now, like it was a regula falsi or seek in regula falsi, see convergence less than one means there was less convergence. What is happening in the secant? If we see it is going like this, then in the secant, this line is one here, whose slope is one, and from above, we can see that the convergence is going very fast. Its slope, its slope is converging, its slope is okay. So from here, we will show that this means this Newton Raphson gives second order convergence. It went from here to here, then came here. So this convergence is this, and the best advantage in this is that give an initial guess — if not, give two. Then from here, we will call the function and here it will check that all the conditions are satisfied. So whoever asks for a function from us, and we will give the function.

In such a situation, I can take the second function. Which function is this? I will take this. I will comment this, and here I will do Control + V. So I have taken the function x square minus cos x. Okay, so its derivative this will have to be defined. So what do I do here? By writing return 2x minus — plus will become plus sin x. np dot sin np because we called it from NumPy. So its derivative of x square becomes 2x. And of cos comes minus sinx, so, minus will become plus.Now we see by calling this. So we have iteration, and see, in the third iteration, we got its root. And see the error — you get minus 1 in the first step, then -2 in the second step, and -4 in the third. From this, it will be known that how fast the iteration is happening in it. Okay, so even in three iterations now its solution is done.

(Refer slide time: 56:26)



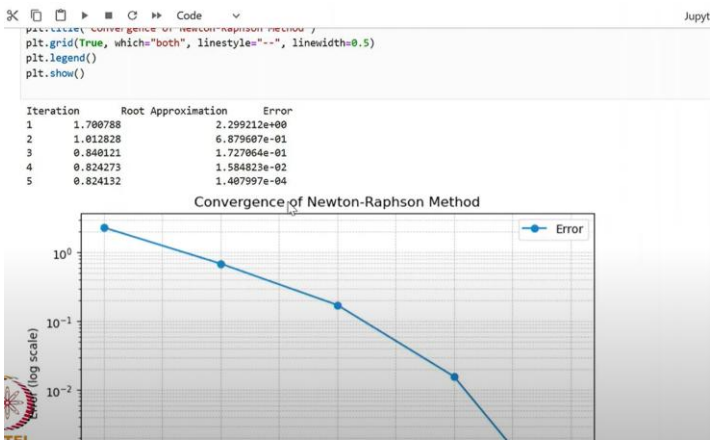
So we can see from here that in the first command which we started with, in the first iteration we got an error of 10 raised to power -1, then in the next iteration it became square of it, so it became -2, in the next iteration it became -4.

So from here, we can find out how fast the convergence is coming. We did the same thing in the secant. So see how much the errors were in the secant — it was -1, then it remained -1 only. Then after this, -2 remained, then it became -4, then it became -6. So from this, we are getting to know that the convergence is close to 1.6. If I see the same thing in this, then it is coming linear — see -1 was 10 power -1, then in the next, it was also 10 power -1, in the next it was also 10 power -1. So from here, it is getting to know that our convergence is linear. So it is coming linear convergent. So from this way, we get to know that our method is — what is its order of convergence?

So the Newton Raphson method gives fast convergence. And now if I write zero in place of its initial, then this derivative is zero, right? So what does it mean?

Its derivative is zero, so we cannot take zero. I take four. Now let's see. Now the iteration has come, so as soon as I wrote four, I took 1.71 from there and then it converged and came to this.

(Refer slide time: 58:08)



So see, one, one, then two, then four. So from here we came to know that the order of convergence of this method is second order convergence. So from here we have shown Newton Raphson method for second order convergence.

Now what we do is, let's see the method of 2D. So we took Newton Raphson method for 1D. Now what I do is check it in 2D. Now what happens in 2D will be a little complicated, but it will have to deal with vectors. So to deal with vectors, we have to use capital F, which is our function, J, as I told you, is the Jacobian, x0 I told you is the initial point, this is the tolerance, so I write the tolerance like this and the maximum iteration. So this Newton Raphson is the same thing. It will be for two non-linear equations. This is fine and everything will remain the same. We have put the values. It is just that now two values will have to be defined in it — F values — so an array will have to be defined because there will be a vector and J is also a matrix. So the array is formed. So we know how we defined the matrix, how we defined it, we have learned in Python. So what will we do from there? We will create these values. After that, what are we doing now? np.linear algebraic is an inbuilt software,. it ia a module. We are using it to solve this. So what are we doing now? We are solving J value minus F value. So this is what I told you — that J will have to be multiplied, right? So solve this system of questions. And the x next is x plus delta. This delta will come here. So this is our Newton's Raphson formula.

(Refer slide time: 1:00)

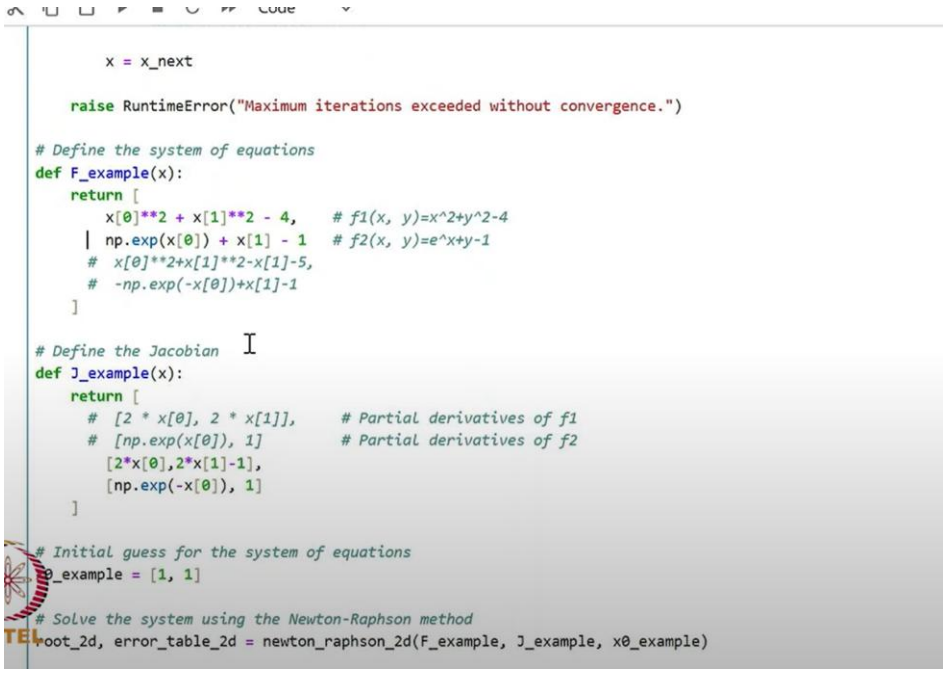
```
    Returns:
        x (numpy array): Approximated root [x, y].
        error_table (list): Error and root approximations at each iteration.
    """
    x = np.array(x0, dtype=float)
    error_table = []

    for i in range(max_iter):
        F_val = np.array(F(x))
        J_val = np.array(J(x))

        # Solve J * delta = -F
        delta = np.linalg.solve(J_val, -F_val)
        x_next = x + delta
        error = np.linalg.norm(delta)

        error_table.append((i + 1, x_next[0], x_next[1], error))

        # Check for convergence
        if error < tol:
            return x_next, error_table

        x = x_next

    raise RuntimeError("Maximum iterations exceeded without convergence.")

# Define the system of equations
def F_example(x):
```

After that, we will calculate the error. We will keep writing that error here. x next 0 1 means its first coefficient, this second coefficient, this x coefficient, this y coefficient. And this is our iteration and this error. So it will keep going on like this.

Now the main thing is what we have to do — how to define the function. So see, how have I defined the function? So in this case, I have taken x square plus y square minus y minus 5. I have taken this from here because x0 is there. Now, let me first take this. And I have taken this. So, look, I am taking the function x square plus y square minus 4 — so, x square means the zero of x, the first element is its square. Then the second element is its square minus four.

Because there are only two elements in this, what is f2? So, exponential, so np dot power of exponent x. So x0 plus x1 minus 1. So, this is what we have. This is the function which we have defined.

(Refer slide time: 1:01:50)



Now I need the Jacobian of this. So what do I do for Jacobian? From here, I define the Jacobian of this. Jacobian means matrix. So what will come? It's Jacobian will come first, 2x comes on this. The first value is in the first row. Now in the second value, 2y comes in the next row. So this means we have one row. The e power of x of the second row will remain the e power of x. So for np dot exponent x0 and for y it comes 1 so, 1. So what did we do? From here we have an initial guess, which means we have defined the Jacobian, and we have defined this function.

Now what did I do? I took the initial guess. Okay, so we will take the initial gas. Let us see what value comes. So in this case, if I run it, then we got these values. So I calculated it. So in the first iteration, x became -1 something y something and the error is this. So like this, we had the iteration — eight iterations — and in the last we saw that our errors were: the value of x came out to be this, the value of y came out to be this, and the error is one, became square of two, then three, then seven, and then squared — exactly 14. So it has reached here. So how fast has this convergence gone? It has converged exactly. So in eight iterations, we got its roots, and its root came out to be this. Okay, its root came out to be this. So we can change it like this.

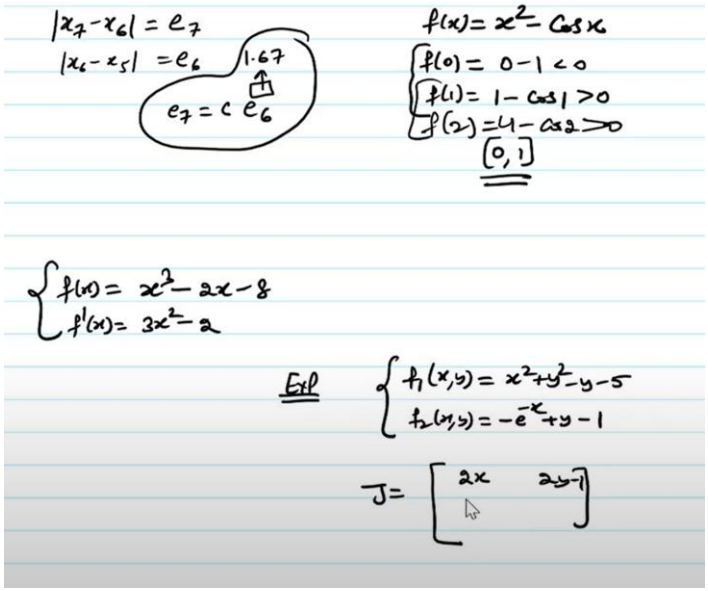
So I had taken this function. I have just defined this function. Now, suppose I take another function that I have defined. I have taken x square plus y square minus y minus 5, and the second one is the e power of x minus y minus 1.

(Refer slide time: 1:04:20)

```
        x = x_next

    raise RuntimeError("Maximum iterations exceeded without convergence.")

# Define the system of equations
def F_example(x):
    return [
      #  x[0]**2 + x[1]**2 - 4,    # f1(x, y)=x^2+y^2-4
      #  np.exp(x[0]) + x[1] - 1   # f2(x, y)=e^x+y-1
        x[0]**2+x[1]**2-x[1]-5,
        -np.exp(-x[0])+x[1]-1
    ]

# Define the Jacobian
def J_example(x):
    return [
       # [2 * x[0], 2 * x[1]],     # Partial derivatives of f1
       # [np.exp(x[0]), 1]         # Partial derivatives of f2
         [2*x[0],2*x[1]-1],
         [np.exp(-x[0]), 1]
    ]

# Initial guess for the system of equations
x0_example = [1, 1]

# Solve the system using the Newton-Raphson method
root_2d, error_table_2d = newton_raphson_2d(F_example, J_example, x0_example)
```

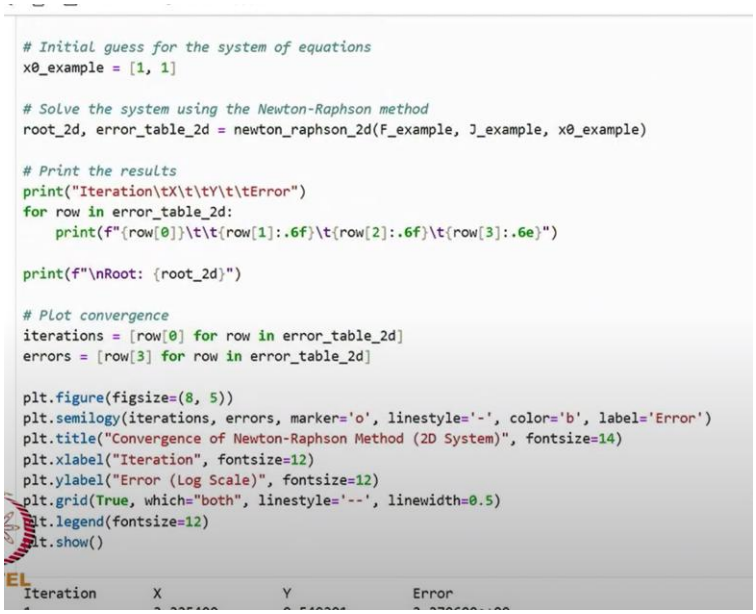
So if I write it here, I am taking this function — this one — and f1(x,y), which I have taken. I have taken x square y square minus 4. And at f2(x,y), which I have taken, I have taken exponential plus y minus 1. Let me see. I have taken this function. Did I change anything? I have taken plus x minus 1. I have not taken x square plus y square minus y minus 5. We took e power minus x plus y minus 1 So we took this. Okay, so the jacobian that comes out of this — we can take out the jacobian from here. I had just defined 2x, 2y minus 1 . Okay, so we did it like this.

(Refer slide time: 1:06:06)



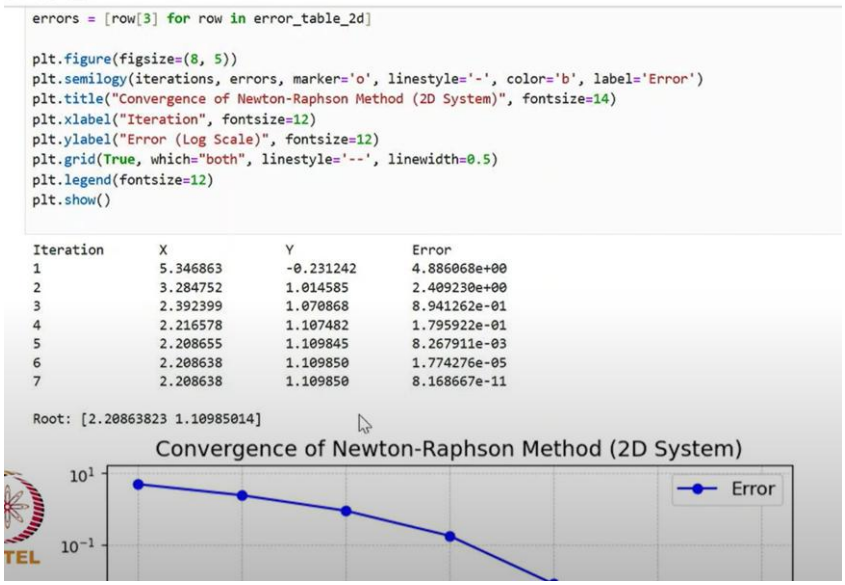
So if I look at it, here comes our 2x, 2y minus 1 and exponential minus this and one. Okay, so I had just done it. So if we want to calculate it, then I will take its roots from one one and let's see what comes out. So we initially took one point, and see, in the sixth iteration, we got its roots , 2.20 and 1.10 and this is the convergence. You can see that in the first iteration, there was this much error, then it became minus 1. Then in the second, it remained minus 1, then its square became minus 2, then its square became more than square minus 5, then the square became minus 10. So from here, we will get to know that the error is giving its second order convergence. Okay.

(Refer slide time: 1:07:05)



So, what we do in this is that we can change the initial value. It is not that it will come only like this. If I write zero, then I took zero. So, see, after nine iterations, it has come to us. It has converged to another root. We have to take one. Let's take two. Now, let's see. Yes, it has come beyond 2 and 1. Its convergence is 2.20 and 1.10.

(Refer slide time: 1:07:49)

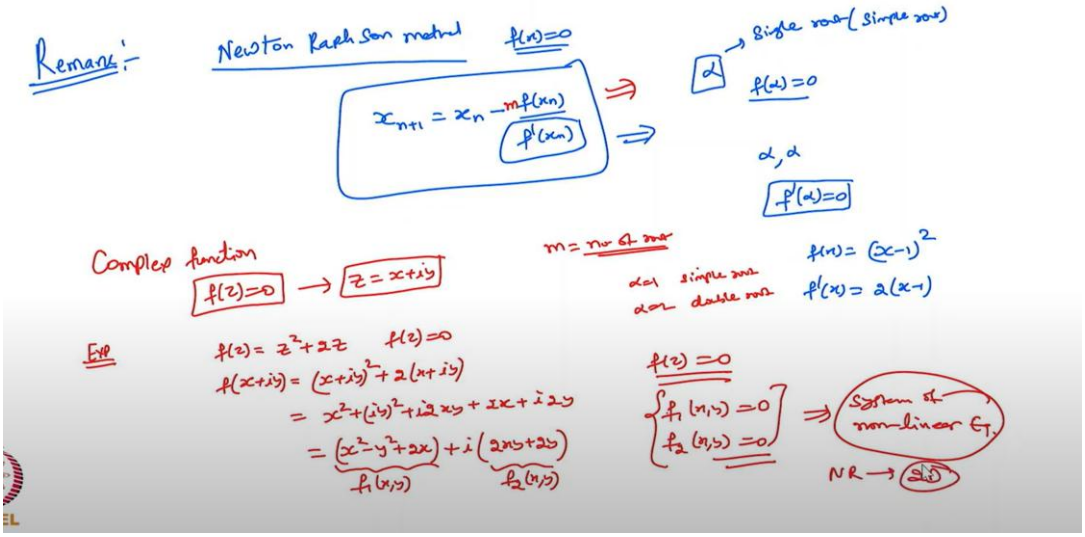


It has converged because it is nonlinear, so it can have many roots. So, it will converge to that root. If I take two two, then let's see what happens. Even then, it is converging to the same. Even in five iterations, it has converged. So, we can play with these codes and find out the solution.

So, this Newton Raphson method is quite useful in the case when we have one-dimensional and two-dimensional problems. Even which is our Newton Raphson method, if you look at it, we can use it for complex. If we want to use it for complex, the question is how we can use it for complex. See, if we have a complex given f(z)=0. So, we know how complex functions are defined. So, we have f(z)=0, where z is a complex number. Now, what do we have to do? Find the root of this. So, what would the root be? If we want to find the root of this, what would we do? If we want to find the root of this, what would we do? For example, let's take an example. For example, we have f(z) is given. I have z square plus 2z. This is the value. So, we were told, brother, find the root of this, where this z is complex number. So, if you see I can write this as: x+iy so this becomes our x+iy whole square plus 2(x+iy). Now, I can square this x square plus iy whole square plus 2 xy I plus 2x plus I 2y. Now, what would we do? We would separate the real part and the imaginary part. So, this would become minus y square. So, I can write its real part x square minus y square plus 2x. This is plus i 2 xy plus 2y so, this has come.So, here 2xy and here 2y. And here, we have this. So, you see, if we want to solve this, we would take the real part.

We will take this. This is the real part. I will name it as f1(x,y) and I will name it as f2(x,y). This is the imaginary part. So, if we have to solve f(z)= 0, then basically what we have to solve isf1(x,y)=0 and f2(x,y)=0. And what is this? This we have just solved. This is the same as system of non linear equations, and we have just solved this by Newton's Raphson in two dimensions.

(Refer slide time: 1:11:50)



So, if we have to find the roots of the complex function, then we can use Newton's Raphson method and we can easily find it out. So, this is beneficial for us.So, today we did a very important method, which was Newton's Raphson, and we saw that Newton Raphson is second order convergence, and we can very easily extend it to higher dimensions.

So, I hope you have understood this lecture, and thank you for watching it.