Scientific Computing Using Python

Professor. Vivek Aggarwal and Professor. Mani Mehra

Department of Mathematics

Indian Institute of Technology, Delhi

Lecture No. 08

Welcome all of you to Scientific Computing Using Python. So, in the last lecture that we did, we saw how the round-off error or chopping-off error evolves — what is called a significant digit. So today we will go a little further from that and do the arithmetic of errors. Let's start.

In the last lecture, we had worked out the relative error. We had worked out the percentage of error. So today we will do arithmetic operations. We have to check how the error involve, increases or decreases, we have to check this. So, what does operation mean? For example, if we have two numbers, take x1 and x2, these numbers which we have rounded, which we have rounded to x1* and x2*, okay? Let ε_1 and ε_2 be the corresponding errors in x1 and x2 What does it mean? $x1 = x1^* + \varepsilon_1$ and $x2 = x2^* + \varepsilon_2$. Because we know that $x1 - x1^*$ is an error and we also checked that this error is called rounding error. We rounded it off, so that's why the error is there.

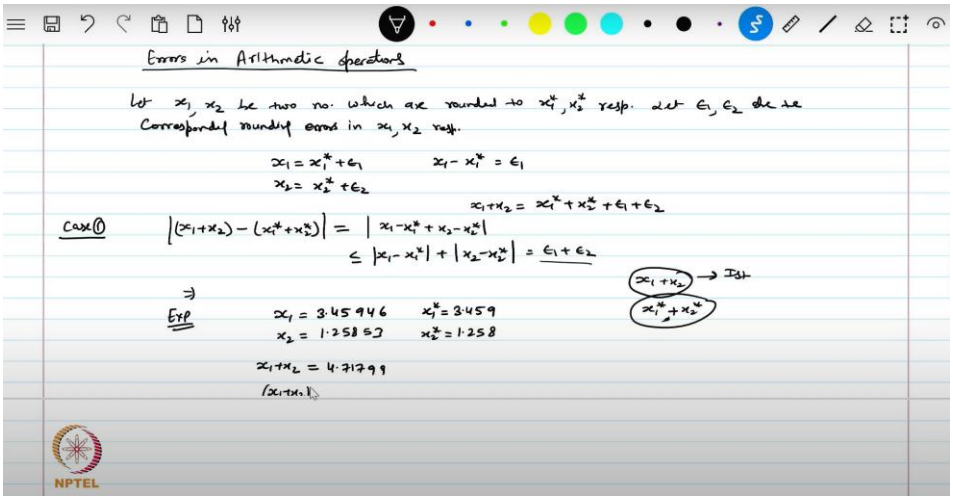
Now what do we have to do? So, let's take the first case. Now we have to check that, suppose we have sum two numbers, so what will happen after making them sum We have to check how the error occurs in x1 + x2 and in x1* + x2*. Because we know that $x1 + x2 = (x1^* + \varepsilon_1) + (x2^* + \varepsilon_2)$. We wrote it here like this. So, we want to see what will be the errors between the two. We can write it like this: $|x1 - x1^* + x2 - x2^*| <= |x1 - x1^*| + |x2 - x2^*| = \varepsilon_1 + \varepsilon_2$. We wrote it like this and we know that these are errors: $\varepsilon_1 + \varepsilon_2$. So, from here, we get that the errors which will evolve in this case will be smaller than the values x1 and x2.

What does it mean? That we first performed the operation x1 + x2, and then after that we rounded off the summation that we got. We did not do that, we rounded off x1*, then rounded off x2*, and then took the sum of them. What will happen if we do this is that the error will increase. And if we do not do this, first add x1 and x2, and then we do the rounding then see that the error will be reduced. So, the error that will come in the summation will be less than the corresponding error when we had done the rounding earlier.

So, this will mean that we should first perform all the operations in the computer, and then at the end, where we want to see the result, we should do the rounding there. We should not do that we keep on rounding at every step. What will happen in that case is that the error will increase a lot. So, this same case and similar case will come in subtraction too.

Now if I take an example: Suppose I took x1 = 3.45946, okay? And I rounded it off and I got 3.459, okay? So, I kept it up to three decimal. I rounded it to four decimal places. I took x2 = 1.2853, we took this number. And x2, which we rounded off, came to 1.258 because it is an even number and there is a five here, so it will remain like this.

(Refer slide time: 07:09)



Okay, so now what did we do? We summed x1 + x2, so the sum of these two came to be 4.71799. This will come, okay? Now, after summing them, I rounded them. Okay, so I rounded it like this. Now what did I do? I summed x1 + x2, and I rounded it. So, after rounding it, the value that I have is — if we round it to a three decimal, we get 4.718. Okay, so we just did it like this. We did not do it in normalized form. We were doing it just to check.

So, we found this value. So, we got the rounded number of this x1 + x2. Okay, so now what have we done? |(x1 + x2) - (x1* + x2*)|, let's check their value. So, if you see, the value will be 0.0001 because if we see this value, it will become zero here. It may come out negative, no problem. We are taking the summation. This value is zero, okay?

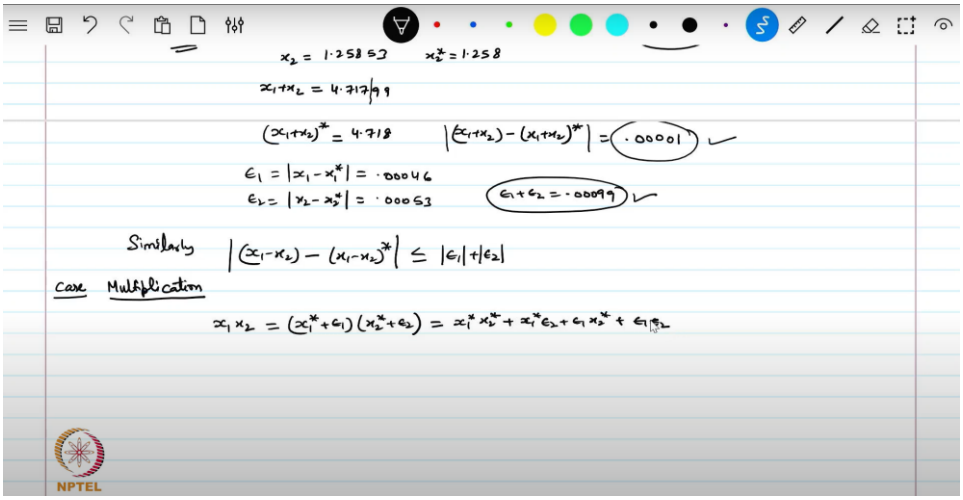
And from here, we got to know, and if we find out how much is |x1 - x1*|, and if we try to find it, then we will get 0.00046, and the error for x2 will come as |x2 - x2*| = 0.00053. So, if we add both errors, it will come to around 0.00099.

So, see what we did: the summation error would be this much. And if we did it after combining, then it would become this much smaller. So, from here we can say that this error is very much smaller than this error. And this means that this error would be much smaller than the individual error. So always, we have to round off only after completing all the operations that we have to do, right?

So, this is the same thing which we can do in subtraction also. Subtraction and addition are the same. So similarly, x1 – x2, we take the values of x1 – x2 and round it off. Then we will subtract this and also get similar behaviour the error will be smaller if rounding is done after subtraction.

Or we can define its value like this also. So here, I have taken just the value of |ε1|+| ε2|, so that's why we will take the model as the value, right? So, we got this. This thing happened similarly in subtraction.

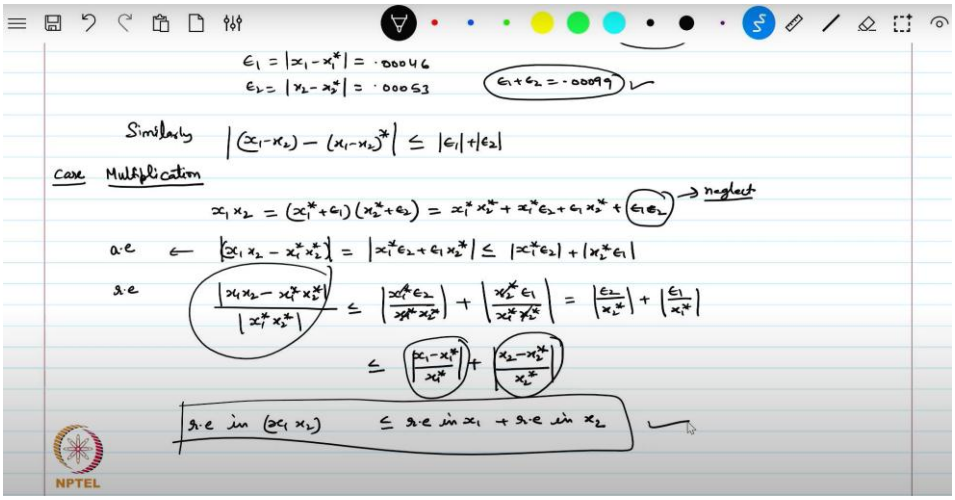
(Refer slide time: 11:59)



Next, our case is multiplication. So, in multiplication, what is happening? What will we do now? First, we took two numbers, x_1 and x_2, multiplied them, and after that, we will do x1 into x_2. So, what is x1? Rounding plus error. x_2 plus ε2. If we expand it, then we will get x1* × x_2* + x1* × ε2 + ε1 × x_2* + ε1 × ε2. Now we have it. Now see, the values ε1 and ε2 are very small values among themselves, and the product of these two will be very small. So, what will we do? This quantity we will neglect, because it will be a very small quantity, so it will not have much effect on it, so we will neglect it.

Now what do we do? Let us try to find out x1 × x_2 minus x1* × x_2*. So, I took its modulus, okay? So, we brought it like this. So, it became x_1* × ε2 + ε1 × x_2*. This value came to us. I added it, and if we solved it, then we will get |x_1* × ε 2 + x_2* × ε1|. And this is triangle law we know, so we can do this. Now, what do we have to do? We are not getting any information from here. So, what will we do now? We will try to find the relative error because this is an absolute error, right? So, this is an absolute error. We did above also, that was an absolute error. So, this also we have written in short form, but it means absolute error, which takes the absolute value.

(Refer slide time: 16:44)



Okay, so now what do we have to do in this? We will try to find the relative error r.e, means relative. So, what will we do in relative error? We know the formula. So, we will divide the error, which is | x1x2 - x1*x2*| minus the absolute, by the product x1* × x_2*. So, we will

have to do both sides. Now what are we doing? This also has both sides' product. So, this is divided by x1* ε_2 /x1* x_2*. So, we can write it like this: ε_2 over x_2* + ε1 over x1*. So now we have this value. So, we can take it like this.

See, what is x1? x_1 is x_1 -x1*, and I divided it by x_1* plus x_2 minus x_2*, because it is ε_2, and I divided it by x_2*. So now we get to know what this is. This is the relative error in x_1, and this is the relative error in x_2. So, we can write this from here, that this will be less than or equal to relative error in x_1 plus relative error in x_2.

So, what we get from this is: relative error in $x_1 \times x_2$ will be less than or equal to the sum of relative error in x_1 and x_2. Okay, so what does this mean? The maximum relative error of the product of two numbers will always be less than or equal to the sum of their individual relative errors. So, this is what we get. So, what does it mean?

First, we multiply x_1 by x_2, and then do the product. Then whatever value comes, if we round it off, the error will be very less — compared to if we round off the numbers first and then do the product. So, this is the relative error that we have. So, we have done this in product.

Similarly, we can do it in division as well. So, the next case that comes is of division. So, what did we do in division? x_1 divided by x_2. Obviously, x_2 is not zero — only then we can take the division. So, we will write it. So, we can write it like this: x1* + ε1, and x_2* + ε2. This is what we have. So now what do we have to do with it? I will do it like this — I will write x_1* common. So, I wrote 1 + ε1 over x1*. I divided it by x_2*, 1 + ε_2 over x_2*. This is done.

So now what do I do with this? x_1* divided by x_2*. This is here, and on top, I will be left with the quantity 1 + ε1 over x1*. I take this up as well, and this becomes our (1 + ε_2 over x_2*) $^{-1}$. We can expand this. So, after expanding, we will get this quantity. So, this is now with us. If we put a minus on it, then it will become 1 + ε1 over x1* into (1 − ε_2 over x_2*), just take it till here and the terms that will be of higher order, we will neglect them.

Okay, so we will write: we are neglecting higher power of ε_2, because after that square will come — we will neglect it. Right? Now this comes to us. So, we will expand it. Now we multiply it. So, we get this: x1* over x_2*. Now this comes to one, okay? Plus. Let's take 1 − ε_2 over x_2* + ε1 over x1* − ε1 × ε_2 / (x1* × x_2*). This quantity will come. Now this value has come to us. So, we will neglect this as well, it will become a very small value.

What does it mean? Now, only this quantity will remain with us. So, we will take this and neglect it. So, we will take it to the left side. So, we got the quantity from x1 over x_2 − x1* over x_2*. This quantity if we take it to the left, the quantity that we have left now, we will take it to both sides. So this is left: ε1 over x1* − ε_2 over x_2*. This quantity is left with us.

Now what do I do? We took its modulus on both sides. So, I can write it like this: | ε1 over x1*| + | ε_2 over x_2*|. This value is right. So now what do we have to do? This is with us — absolute error. This is absolute error, but we have no value from here. Information is not being found in the sense of product. Okay? We can only say that the relative error which was in x1 and relative error in x_2 — and the relative error in the product will be less than these.

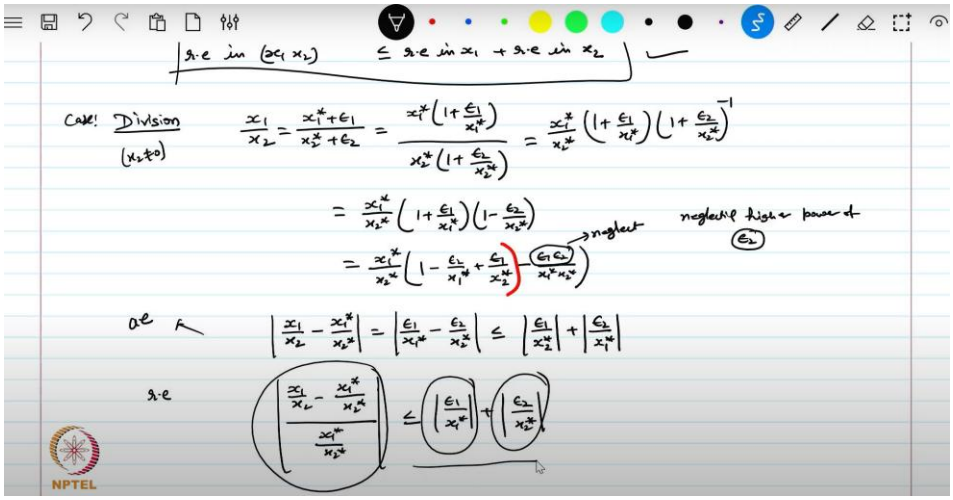
So, we can say from here that the absolute error of the product is correspondingly in this. What is it from x1? Then x_2 will come. Okay, we will have to change it a little here. If we product both the quantities in this, then x_1 will cancel it, then ε1 will come over x_2. So, I will

correct it a little bit. This and this are okay. So, this becomes our x_2, and this x_1. Okay? So, x_2^* is coming below $\varepsilon 1$, and x_1^* is coming below at. Because if we take it inside, the terms will get cancelled.

So, what does it mean? That we will not be able to make any conclusion from here, because inside this, this x_2^* will come and this $\varepsilon 1$ *. So, with this formula, we will not be able to do anything in the absolute error.

So, what will we do? If we find the relative error, then I found the relative error and divided it by $x1^*/x_2^*$. So here we will do both sides. So, if we do both sides, then you know that here x2 will cancel out from x_2. Here the quantity will come: $\varepsilon 1$ over $x1^*$ + ε_2 over x_2^*. So, after doing this, now we know that the relative error which was there in the division is less than or equal to the relative error in x1 and the relative error in x_2.

(Refer slide time: 23:47)



So, if we add, subtract, do multiplication, and do division, then we know that the error which involve after that, if we take the relative error, then it will always be less than the individual error. So, in this way, we can add the terms, we can subtract, we can divide. So, these are the operations. We will apply these things in the computation that we will do. Keep in mind that after we have done all the computations first, in the end we will round off the whole numbers.

Now I have told a little about the error—how the error occurs. Similarly, we also have truncation error. So, what happens in truncation error is that, like we have a series, right? Now, I am using a function—exponential. So, the exponential series is like that. This is our exponent series.

Now, what are we doing? I have to find out the value somewhere—say, I have to find out e2 or e5. So, what is happening in the computer? How are these values getting calculated? With the help of the series, because we cannot find out the exact value on the left-hand side.

Now if someone asks me what is the value of e2, then how will I find out the value? What will I do? I will keep 2 in the exponent. That is, 1+2+2^2/2! +2^3 / 3! +…. like this, and from there I will calculate the values. And we will get the value of power two of e.

So, what do we have to do now? If we have to find this value in a computer, then what will the computer do with it? Now, we will not take an infinite number of terms because that is

not possible. And if we take a very large number of terms, then that will also use a lot of computation. So, what will the computer do? It will truncate the series . For example, it goes only till 2^n/(n+1)! and truncates the entire series till a particular term. The next values will come in the form of error. So, the value we will get is approximate sum of the series, and the truncated terms are left out.

Now, the value we will get is called the approximation, and the remaining part is the error, or the remainder term. Suppose I defined the approximated value as a, and the remaining terms as r. So, the actual value of e2 becomes a+r. Here, a is the sum of terms we computed, we have fixed It like 10 terms or 20 terms, and r is the remainder (truncation error).

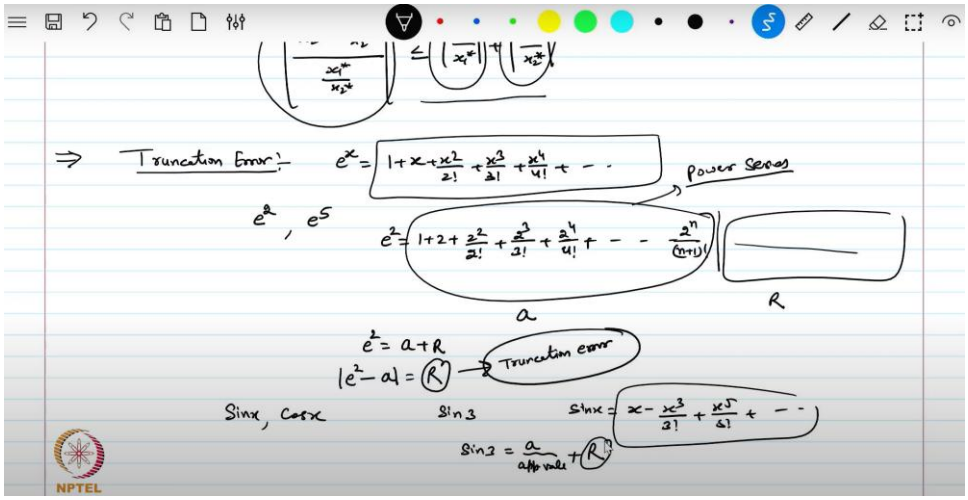
If someone asks how much error has occurred, this truncation error will be r=|e2−a|. We generally take the modulus to keep the value positive. So, this r we will call as truncation error.

Similarly, we often use trigonometric functions, like sin(x), cos(x). Now, if we have to find the value of sin(3), we don't know the exact value. But we do know the power series of sin(x), i.e.,

$$x - \frac{x^3}{3!} + \frac{x^5}{5!} \ldots\ldots$$

This is called a Taylor series or power series. We know all these power series. So, if I want to find the value of sin(3), we take a few terms of the series, substitute x=3, and get an approximate value a plus the remaining portion r becomes the truncation error.

(Refer slide time: 29:07)



So, in computing, whatever values we are calculating, a truncation error always involve if we are using series or transcendental functions. For example, if we calculate e^0, it is exactly 1, and no error occurs. But for any other value of x, we need the help of the series, and thus, a truncation error is involved.

The remainder r converges to 0 as we increase the number of terms in the series. Let's give it a number n, so as n increase the remainder term will tend to 0. So, truncation error plays a major role in computational error. Therefore, in computational errors, we cannot avoid rounding errors or truncation errors.

Now that we've talked about errors, let's start solving non-linear equations using scientific computing or numerical methods. These methods help us solve equations and determine whether roots exist.

Suppose we have a function like:

$f(x) = x\text{^}3 + x - 1 = 0$

We can easily find its roots using algebra. We find one factor, divide it out, get a quadratic equation, and solve it using the quadratic formula. So, we can easily find the roots of this algebraic equation.

Now take another function:

$f(x) = x^2 - \cos x = 0$

This is not a purely algebraic equation because it contains a trigonometric function, which is also known as a transcendental function (i.e., functions that can be represented using power series). Algebraic equations can sometimes be written in power series, but for functions like sin(x) or cos(x) or e^2 or their combinations, the root-finding process becomes more complex. So, these are called transcendental functions and the equations in which it is involved is called transcendental equations. There can be more complicated equations.

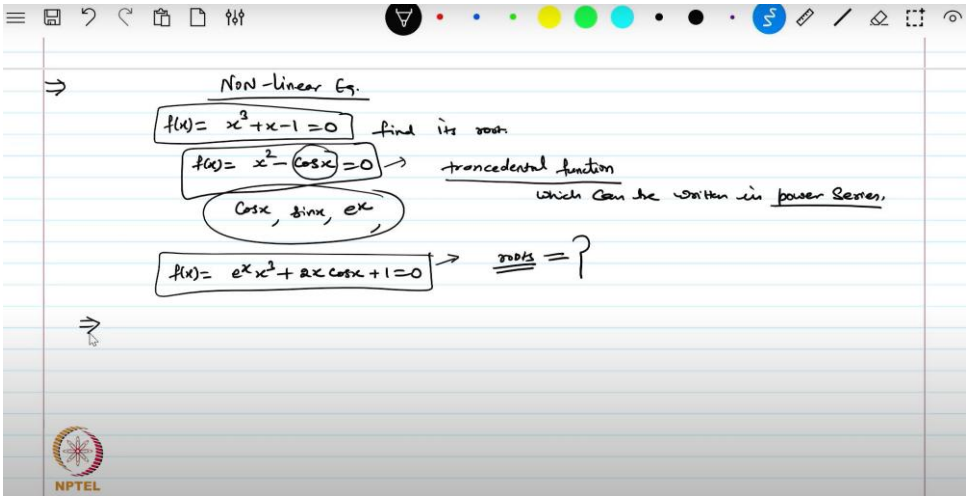
Take another complicated equation:

$f(x) = e^3x^3 + 2x\cos x + 1 = 0$

This is a transcendental equation, and solving it using pen and paper is extremely difficult. We can only guess where a root might be, but we can't solve it analytically.

To find such roots, we need the help of a computer. But before using the computer, we need an algorithm—a clear method that we can program to find the roots.

(Refer slide time: 35:22)



For this purpose, we will now discuss different numerical methods.

The first method we will use is the Fixed Point Method. Other methods include: Bisection Method, Regula Falsi Method, Newton-Raphson Method. All these methods are categorized under root-finding techniques.

Fixed Point Method is one of the most important ones and we'll start with that. So, what do we do in this? We find out the fix point. So, for that, this is showing that the methods that we use will be an iterative process. Iterative means that now we have to find the roots of any equation, f(x)=0. So how do we do this? Find out its roots. So, what we have to do with the roots is that we will have to apply the fixed-point theorem.

Now let's see what happens. So, the fixed-point iterative method, which is a numerical method, we know that it is a technique for finding the root of the nonlinear equation. So, what it does is transform the equation into an equivalent form. So, what we have to do is, first of all, we have to write it in this form because this is x=g(x). So, when its solution comes, we say that x is a fixed point. What does it mean? That x has not moved—there is x on the left also and this is also its root x, so we call it a fixed point.

So, what will we do with it then? Iterate, redefine the values to find a solution. And this method is straightforward to implement and can converge quickly under certain conditions. Okay, so this method converges. Now how does it converge? We will tell what can be the condition for convergence. That if we have to converge, then what can be the condition? So, what you have to do—steps of the fixed-point iterative method, that if we have to apply that method, then what are its steps?

So, the first thing is to convert the root finding equation into fixed point form, where the function g(x) is a continuous function. The second case is how to choose the initial guess. What do you do? Which is the iterative method—now we will make it. So, see, this method has been formed. How did our iterative method become? With the help of this.

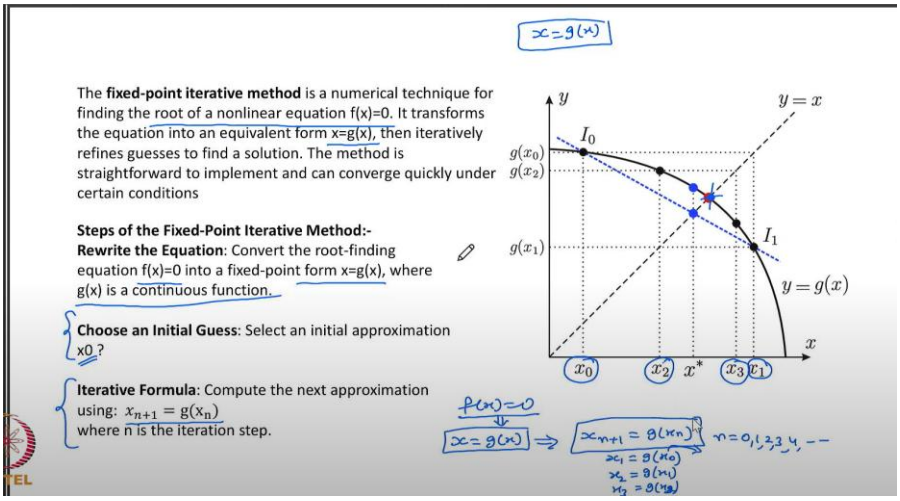
So, what does it mean? Now what did we do? I do it like this—now I have the equation f(x)=0. I had to find its root. I converted it in the form of x=g(x). So, from here I converted it. From here, I went here. I have to solve it with the computer through programming, so I made it an iterative method. $x_{n+1} = g(x_n)$, where n= 0,1,2,3..will keep going like this.

So why is it called iterative method? The iterative method means that it keeps improving itself. We call it iterative method—that we need an initial guess to start. So now, like if I put n=0, then I will get x1=g(x0). Now this x0 is the initial guess which we are saying. I will put its value, I will put the value in the function, which comes to us, from there we will get a new x1.

So now what happens is that the root is—first we assumed the x0 and it got improved to x1, then what will I do? I put x1 in it, so my x2 came here, then I put x2 in it, then x3 came. So now see—like I put x0, then x1 came here, then x2 came here, then x3 came here—like this, you see that this is a fixed point.

Basically, we will call it a fixed point, this one which is red. So, if we have to reach here, then we will keep iterating and after some time we will reach this point. Here, this line y= x and y=g(x), both of them will meet. So, we call this point a fixed point. Okay? So, we will call it an iterative formula for fixed point.

(Refer slide time: 40:40)



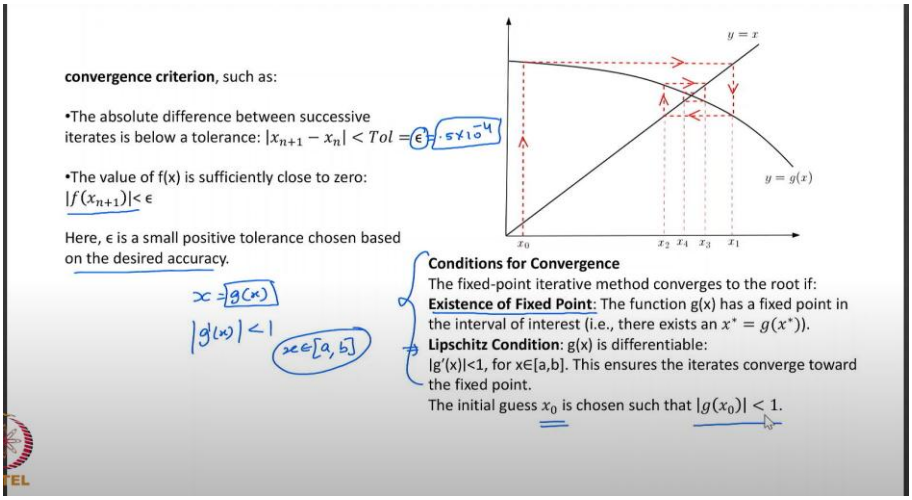
Now how is the method converging or what is its convergence? So, the convergence criterion which we know is the absolute difference between the two successive iterations. Okay? Which is this—we will keep seeing the difference between the two again and again. Okay? And it should be less than tolerance.

Tolerance means—so as I told you, that if I write this here 0.5×10^{-4} , then what does it mean? That the solution I get should be absolutely correct up to four decimal places. So, we will give this value. So. it will keep iterating until we get our solution up to four decimal places. Which means the difference between the two should be the same up to four decimal places. Then we will know that the solution has become, the value of f(x) is sufficiently close to this. So, we will keep finding this value again and again and keep on finding, and we will see that the solution has become very less. Then we stop. We will go and say we have found our solution.

So, see how the solution is converging—first it went to x1, then from x0 to x1, then from x1 to x2, from x2 to x3, and so on. We keep going like this. So, from here, we have got the convergence.. We will give the tolerance from our side. We will check how accurate the solution is. Do we need it to be accurate up to one digit or five digits, or up to four digits? So, in this method, we will give the convergence, that means we will give the tolerance. Now we have the condition. What can be the condition? So, the fixed-point iterative method converges to the root if: the first thing is the existence of the fixed point, right? The function g(x)has a fixed point in the interval of interest, where we think that the root will be found. There, in that interval, there should be a fixed point. It should exist.

Okay? And the second condition that will come—this is the Lipschitz condition of convergence. What is happening in this is that the function g(x), which we will convert to x=g(x), so what are we doing in this? We are taking that the function g(x) should be differentiable. And if we take its derivative, then its value should be less than one for all x belong to [a, b], which means where we can have the root lie. So, this ensures that the iterate converges towards the fixed point. So, this initial guess is chosen in such a way that we satisfy it.

(Refer slide time: 43:44)



Now, for example. Now what do we have to do? We have to choose an example. How to choose g(x)? We have to choose it—so how to do it? And the second thing is that how do we take our initial guess? Where do we see that our initial guess will lie? So, for that, we have a theorem—there is an Intermediate Value Theorem

So, what does the Intermediate Value Theorem say? If we have a function, suppose we have this function, and in it, we take any one value, I wrote it as a, and then we take another value, I wrote it as b. So, this the value of the function at a is f(a), and at b, it is something f(b). So, what did I do? I saw that in f(a) and f(b), if we take their product, it comes out negative. What does this mean? That out of the two values, one is positive and the other is negative.

So, if our function, f(x), says that the function is continuous. Where is it continuous? In the domain. We took the domain as [a,b]. If it is continuous in it, then it says that a value c belongs to [a,b]. That means we will get a value where f(c)=0. What does this mean? It will cross the x-axis somewhere, and that value becomes c. So, this is called the Intermediate Value Theorem.

So, in this, the Intermediate Value Theorem is used a lot to determine the initial guess, because we just discussed how to choose the initial guess. So, the initial guess is here, and now see how it is possible to take the initial guess. This is my function. Now what do I do with it? Let me put zero and check. So, if I put zero, then -1 came up. Okay, if I put 1, then minus 1 came up. If I put 2, so 8-2-1=5, then it came up to 5.

Now see. What does this mean? Now you see that at f(1)f(2), it became minus 5. So, this became negative, and this became positive. So, from this, we came to know that the root of this function lies between them, and if we bring it in between, then we came to know for sure that the root lies in this interval. So now we have to take the initial guess like this.

Okay, so we have to check this. Now what do we have to do? If we want to solve this function using fixed point iteration, then for this, we will have to create it in the form of x=g(x). Now how will we create it? Is it possible? Look, my f(x) was this:

$f(x) = x^3 - x - 1 = 0$

So, someone said that let's take this, I will write it like this:$x = x^3 - 1$

And took x to the other side. So, this is what we have, and from here I said that let's take

$g(x) = x^3 - 1$

Now let's see what will happen. It comes out to be 3x^2, its derivative, right? So, these values will come out to be positive. And if I talk about this interval, then if I put 1 in x, then it will come out to be 3. So, this will be greater than 1. My condition was that it should be less than 1, and only then will it converge. So, what does it mean? That we cannot take this.

So, what can we do? Let's see some more values. Now what do I do? Let me write this differently. Let's take it like this:

$x = (1 + x)^{\frac{1}{3}}$

So, we saw that, let's take it like this. Now what do we do by looking at it? If I write its derivative like this, then our derivative becomes:

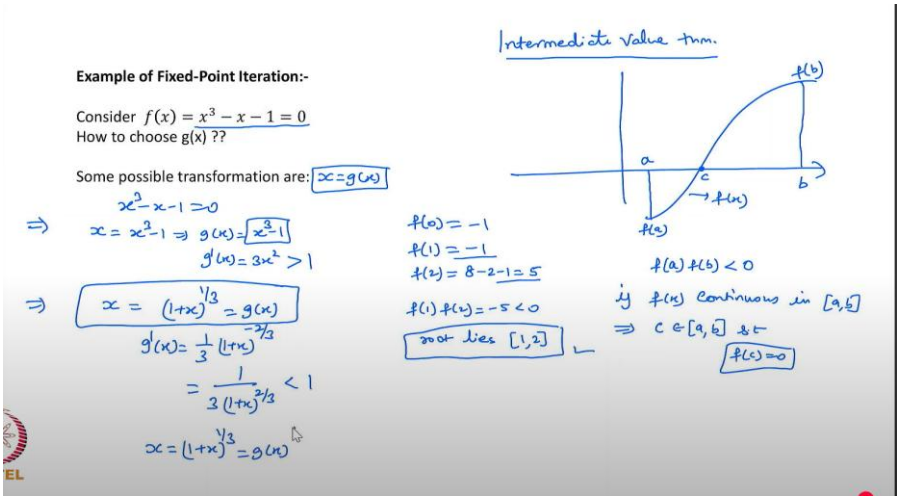
$$g'(x) = \frac{1}{3}(1 + x)^{-\frac{2}{3}} = \frac{1}{3(1 + x)^{2/3}}$$

Okay, so what will we do now? We have this value. Now we have to check—suppose if I put 1 in this, then 2 comes here. Okay. Or if I put 2, then 3. Then this quantity is less than 1, okay? So, what will we do? I will consider this as my iterative scheme, and I will say:

$x = (1 + x)^{1/3} = g(x)$

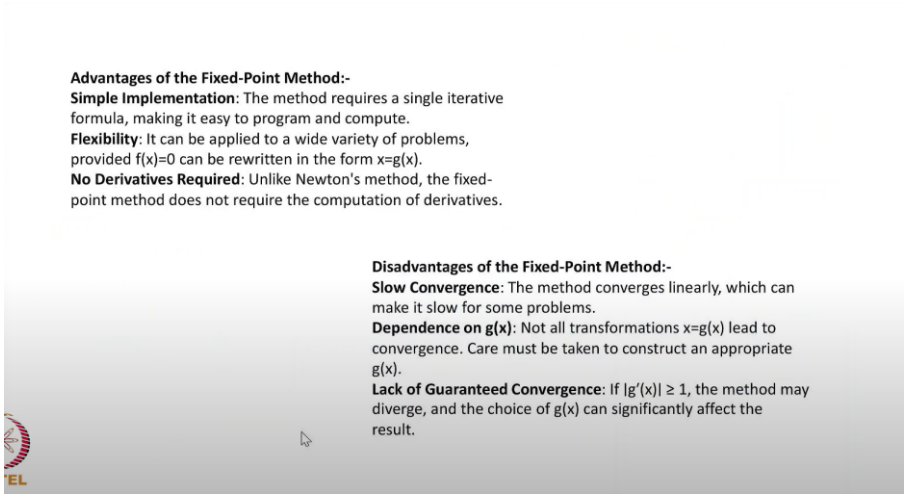
I am choosing this, and my g(x) is, so I will choose this g(x) here so that I can use it. Okay?

(Refer slide time: 50:44)



So now, this quantity has come. Now what do I have to do? I just told you that these values should meet these conditions, only then will it converge. The conditions for convergence. Convergence means that in every iteration, the values and errors should keep on decreasing.

(Refer slide time: 50:53)



So now how will the errors decrease? Let me see. How will I know whether the errors are decreasing or not? So, what am I doing for this? Now suppose we have a root of f, and we got that let x is g(x), and we wrote that:

α=g(α), What does it mean? Alpha is the fixed point. What does fixed point mean? That f(α)=0, which means alpha is the root of it.

So, what did we do? First, we converted the equation into the form x=g(x), and from there, we got the fixed point. Now we also know how to develop a iterative method. I'll write it as $x_{n+1} = g(x_n)$.

Now, if we have to calculate the error anywhere, then we can write the error like this:

εn= α- xn , Here, α is the actual root and xn is the approximation at step n. The error occurs because we are approximating the root, and when we add this error to xn, we reach the actual root α.

Now what can I do? If I subtract from both sides, it becomes:

$\alpha - x_{n+1} = g(\alpha) - g(x_n),\ \varepsilon_{n+1} = g(\alpha) - g(\alpha - \varepsilon_n)$, Now I can write it like this, since g is a differentiable function. Suppose we expand it using a Taylor series, then:

$= g(\alpha) - [g(\alpha) - \varepsilon_n g'(\alpha) + \frac{\varepsilon_n^2}{2} \ldots\ldots]$ and beyond can be neglected if the error is small.

So from here, what do we get? Therefore,

$\varepsilon_{n+1} = g'(\xi)\varepsilon_n$

Now see, εn is the error at the nth step, and εn +1 is the error in the next iteration. We want the error to reduce with each step. So, if we want:

|en+1|<|en||e_{n+1}| < |e_n||en+1|<|en|

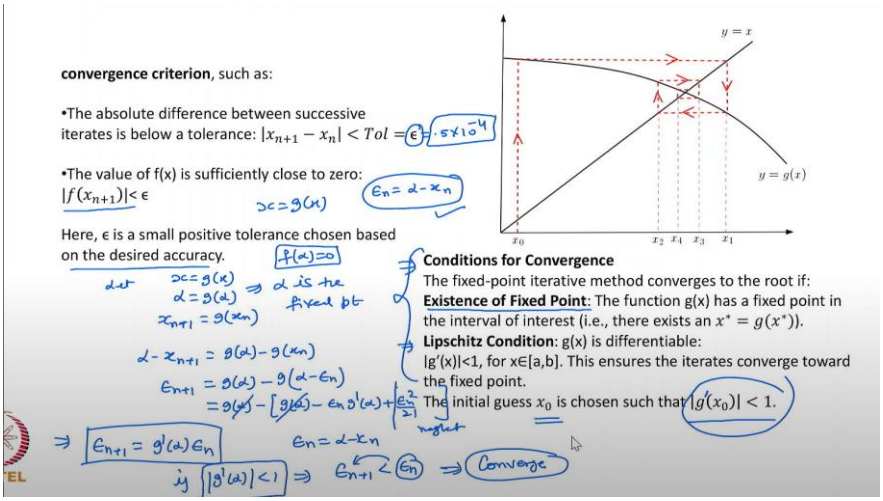
This is only possible if $|g'(\alpha))|<1$ Since $g'(\alpha)$ can be positive or negative, we take the modulus to ensure convergence.

So, if $| g'(\alpha)|<1$ this implies en+1< en , it implies that the error in the next step is less than in the current step, i.e., the error keeps reducing. Therefore, this shows that the method will

converge. This condition is known as the Lipschitz condition or the sufficient condition for convergence.

So this is the sufficient condition: if we choose our g(x) such that it satisfies |g′(x)|<1, then the method will automatically converge, and we will eventually reach the root.

(Refer slide time: 56:00)



Now, some errors might happen during algebra—like in summing or dividing terms—but overall, we now understand how it behaves. After this, we also checked that if we have a non-linear equation and want to find its roots, then we can use the fixed-point method.

We discussed what fixed point is and what its conditions are. In the next class, we will implement Python programming related to this method. Additionally, there is another method, the Bisection Method, which we will also discuss.

So, thank you for watching this.