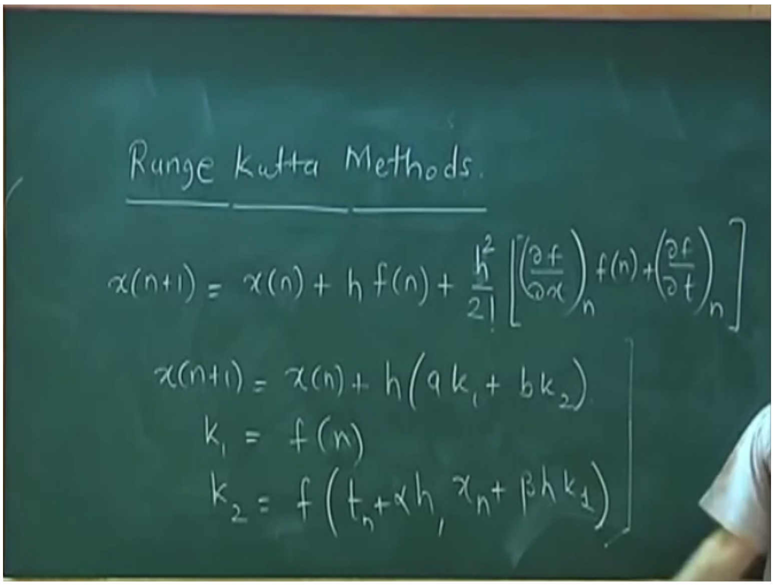
Advanced Numerical Analysis
Prof. Sachin Patwardhan
Department of Chemical Engineering
Indian Institute of Technology - Bombay

Lecture - 42
Solving ODE-IVPs : Runge-Kutta Methods (contd.) and Multi-step Methods

So we have been looking at methods of solving ODE initial value problems and under this category we have till now looked at Taylor series approximations and in Taylor series approximations we said that the trouble with classical Taylor series is computation of partial derivatives. We do not want to explicitly compute partial derivatives. So is there a way out? And these Runge-Kutta methods actually provide a way out of this difficulty.

You can do calculations equivalent to Taylor series approximation without having to explicitly compute derivatives. So you just do function evaluations and the function evaluations are done in such a way that it is equivalent to doing Taylor series approximations. So we looked at last time this second order Runge-Kutta methods.

(Refer Slide Time: 01:25)

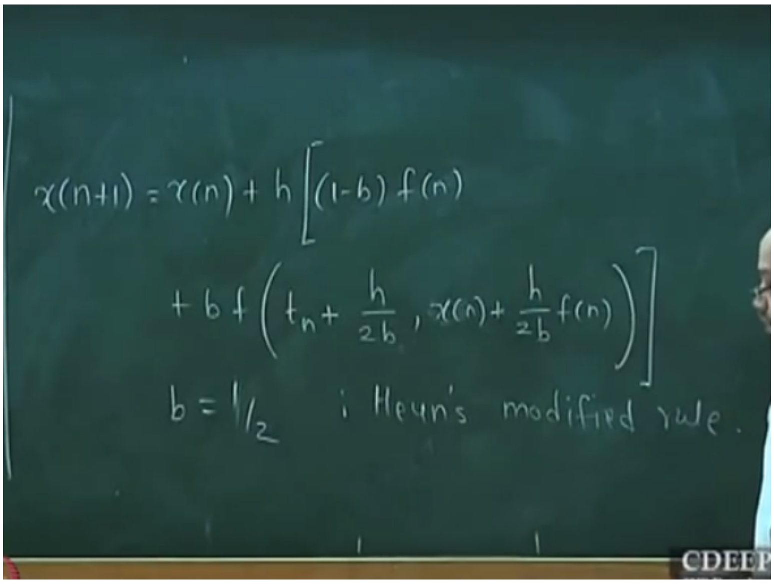


And then I talked about this so general Runge-Kutta method second order Runge-Kutta method. So the classical Taylor series approximation based method would actually have this formula where I would have to compute the first derivative of f with respect to x and t. So this is the classical second order method and Runge-Kutta method tries to do the same achieve the same calculations without actually having to compute.

So we have this Runge-Kutta method xn+1=generic formula I had written was xn+h ak1+bk2 and k1 was nothing but chosen as fn and k2 was. So this is my generic second order Runge-Kutta method, k1 and k2 are function evaluations at some intermediate points and these are carried out in such a way that you match or inspirit you are doing second order Taylor series expansion okay.

So I choose a, b, alpha, beta in such a way that these calculations are equivalent to this formula and then we derived this generic by equating the coefficients doing Taylor series expansion of this.

(Refer Slide Time: 04:37)

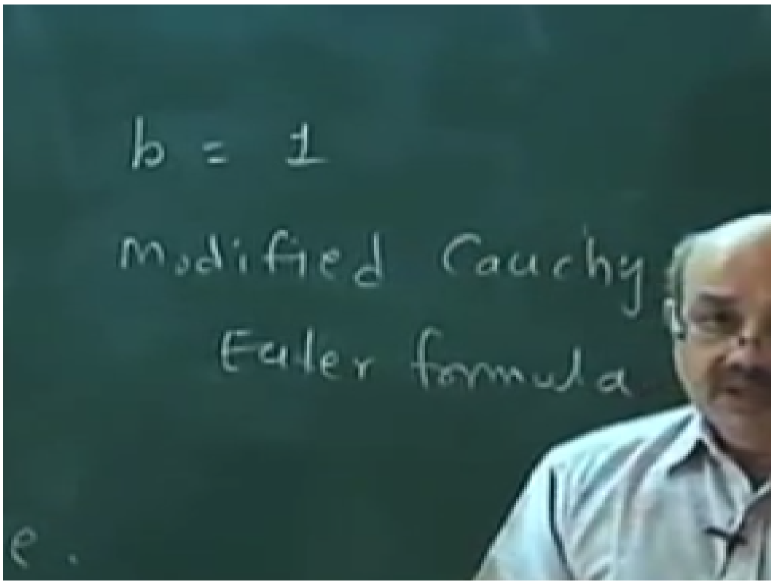


And equating the coefficients we came up with a generic formula, which was so this was a generic second order Runge-Kutta method. Different choices of free parameter b will give rise to different second order methods. So this second order method is actually equivalent to doing this second order Taylor series expansion without having to actually compute these derivatives okay.

So we have chosen function evaluation at intermediate point, 1 in the beginning of the interval and 1 at an intermediate point in such a way that doing these function evaluations is equivalent to doing derivative calculations. So you can think of this as some way of doing derivative approximations using finite points and then rearranging okay.

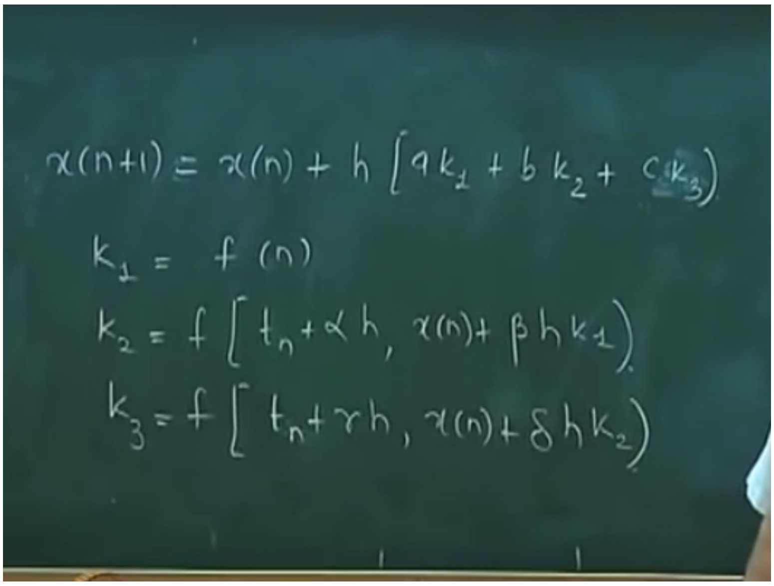
You can do derivative approximations using some finite points and then do a rearrangement to get a formula in which you do not have to explicitly compute. You do not have to compute explicitly derivatives. Now likewise you know and we derived some specific rules with specific formula. We had different choices. We had a choice which was b=1/2 and then that gives you Heun's modified rule.

(Refer Slide Time: 07:10)



Choosing b=1 gives you modified Cauchy Euler formula and so on. So likewise here my third derivative if I want to approximate using third order Taylor series then I will have one more term and then this d3x/dt cube I can expand in terms of df/dx, I will get some complex formula here. I am not writing that right now but I will just give an equivalent Runge-Kutta here.

(Refer Slide Time: 08:20)



So my equivalent Runge-Kutta here would be xn+1=xn+h ak1+bk2+ck3 and k1=fn, k2 is f okay. If I want to approximate a third order Taylor series method using an equivalent Runge-

Kutta method, the way it is done is something like this. Now if you see here you can actually make out a pattern the k1 here is always at the initial point okay. The k1 which is calculated is used here while calculating k2 okay and this is at some intermediate point between 0 to h. The k2 which is calculated here is used in calculating k3 okay and the unknown parameters a, b, c, alpha, beta, gamma, delta okay.

There are 7 parameters to be picked up okay. These you can get by equating the coefficients, you do Taylor series expansion of this, you do a Taylor series expansion of this okay and then equate the coefficients of the terms there. When you equate the coefficients of the terms, you will get typically an over determined set of equations. You get number of equations<the number of unknowns.

And you can pick some variables arbitrarily and then you can fix those arbitrary variables and come up with the remaining variables. What you will get is set of methods, which are called third order Runge-Kutta method okay and logically you can go on writing fourth order, fifth order. Basically, when you want to derive these equations, you have to be very, very patient, do Taylor series expansions properly and then equate the coefficients to get the equations okay.

What you are doing is essentially doing something equivalent to Taylor expansion without having to compute derivatives okay. So this explains entire class of Runge-Kutta methods, I mean you might keep wondering how did you get those coefficients and why computed intermediate points what is the basis? The basis is that these methods actually try to mimic a Taylor series expansion of equivalent order.

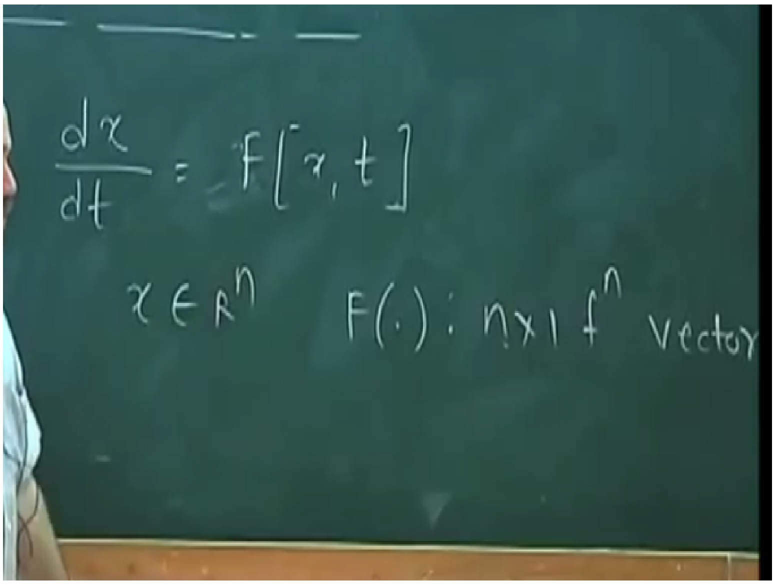
So fourth order Runge-Kutta will try to mimic accuracy of a fourth order Taylor series expansion okay. So that is the basis and looking at this pattern you can go on developing if I ask you conceptually can you write fifth order set of equations, you can okay because k2 is used here. If you have to write for fourth order, k3 will be used okay and 2 more parameters will appear okay.

And then you will get those many equations by comparing the coefficients and then you can solve them. So typically up to fourth or fifth order you might find them coefficients listed. Normally, we can work with fourth or fifth order. You do not have to go beyond that to get.

There is only good accuracy if you choose your integration step size carefully okay. Now what you do when you go for multi-variable method, multi-variable equations?

Actually, we do not do the derivations. These derivations we have done for the scalar case. Right now f here is the scalar of 1 variable equation. We do all the derivations of finding out the coefficients only for the scalar case okay and we simply use that when you go to the vector case. There are no separate derivations for vector case okay. So we just make a simplifying assumption that the same thing will hold and when I have to work with okay.

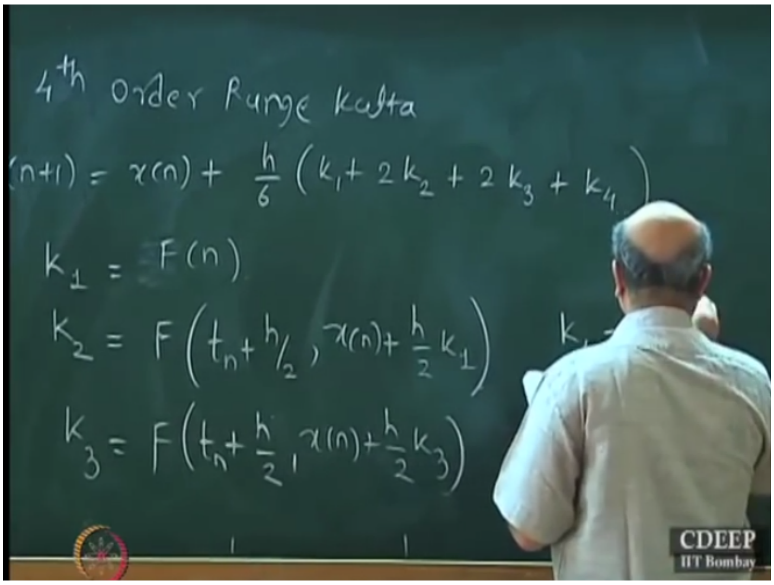
(Refer Slide Time: 13:53)



See for example I want to now do dx/dt=now I am writing this capital F x, t okay x belongs to Rn and f is the n cross 1 function vector right. This is the kind of equations now we actually want to solve. In most of the cases, we have vector differential equations, which are coupled which we cannot separate into separate equations and the real problem is this okay. Now we do not derive coefficients separately for this.

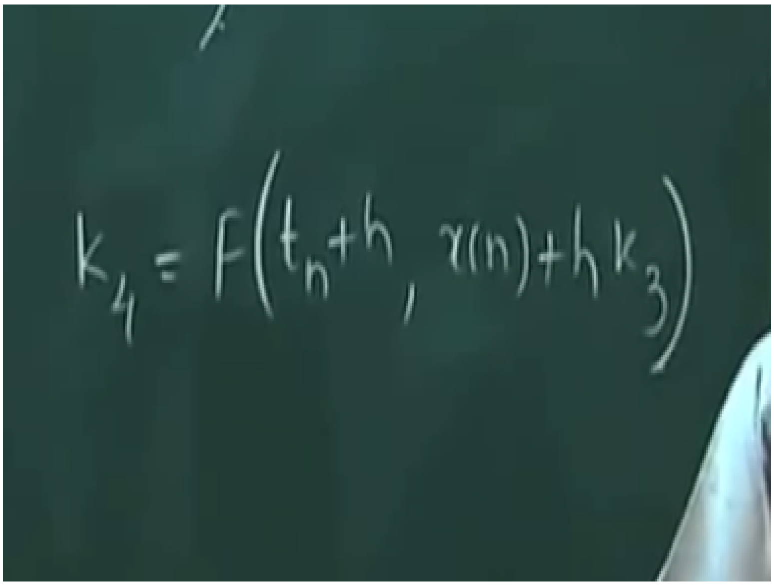
We just derive Runge-Kutta coefficients only for scalar case okay. I will just write down the formula for the fourth order Runge-Kutta, which is using these vector calculations.

(Refer Slide Time: 15:01)



So again you will be able to recognize the pattern and just to give you a feel of what is done so this is fourth order. As you could have guessed, a fourth order Runge-Kutta method will have k1, k2, k3, k4 4 function evaluations okay and the first 1 obviously k1 is just Fn evaluation at the initial point, k2 is F tn+h/2 and mind you this is one way of formulating fourth order Runge-Kutta.

(Refer Slide Time: 16:44)



There will be other ways of getting this alpha, beta, delta coefficients and k4 is F okay. You can notice the pattern, k1 is used in calculating k2, k2 is used in calculating k3, k3 is used in calculating k4 okay. So same pattern and this is multivariate implementation of same Runge-Kutta method. The coefficients here a, b, c are you can recognize the coefficients 1/6. 1/3 1/3 1/6 okay.

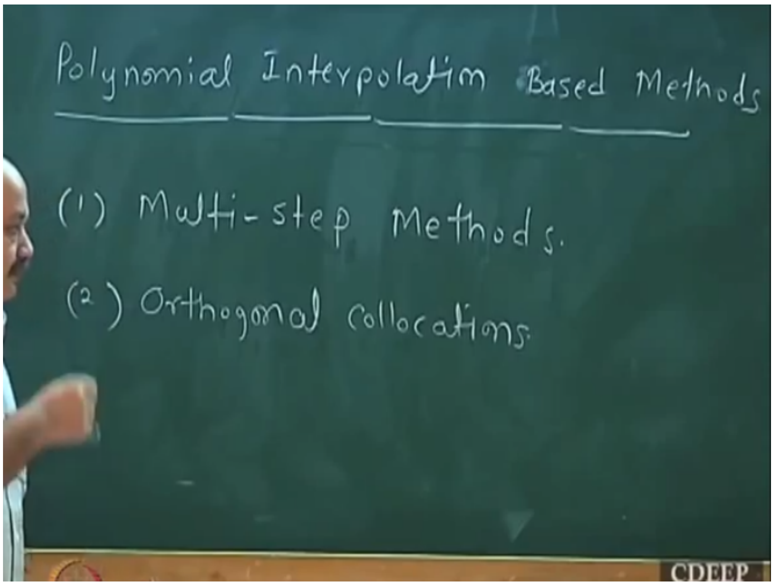
And we can also see alpha, beta, gamma, delta okay and then there are 2 more, 6 coefficients will come here and they have been chosen by equating the Taylor series expansion of fourth order and matching the coefficients. It is tedious but you can just match the coefficients and then choose some of the free parameters, you will get this fourth order Runge-Kutta method. So this is the foundation of Runge-Kutta methods, Taylor series expansions okay.

There are some other variations of Runge-Kutta for example variable step size Runge-Kutta method. I will come to that a little later when we talk about step size selection. How do you choose h? This becomes a very important thing when you are solving ODE IVP and at a later point when I discuss choice of h at that point I will talk about a variant of this called as variable step size Runge-Kutta method.

But for the time being let us now move on to the next class of methods, which is predictor-corrector methods okay. So this is all about Taylor series expansion and its variant which is Runge-Kutta. There are also some variants of this like if you can see this, this is an explicit method okay. The way it is organized if you see here, it is an explicit method because all the intermediate points so this Fn can be calculated, given Fn, k2 can be calculated okay.

Given k2, k3 can be calculated, given k3, k4 can be calculated. So there is no iterations involved here okay that is very, very important. There are some modifications of this which involve semi-implicit some iterative calculations what I have presented here mostly the popular methods of Runge-Kutta are explicit methods okay. The next class that is polynomial interpolation based methods.

(Refer Slide Time: 19:43)



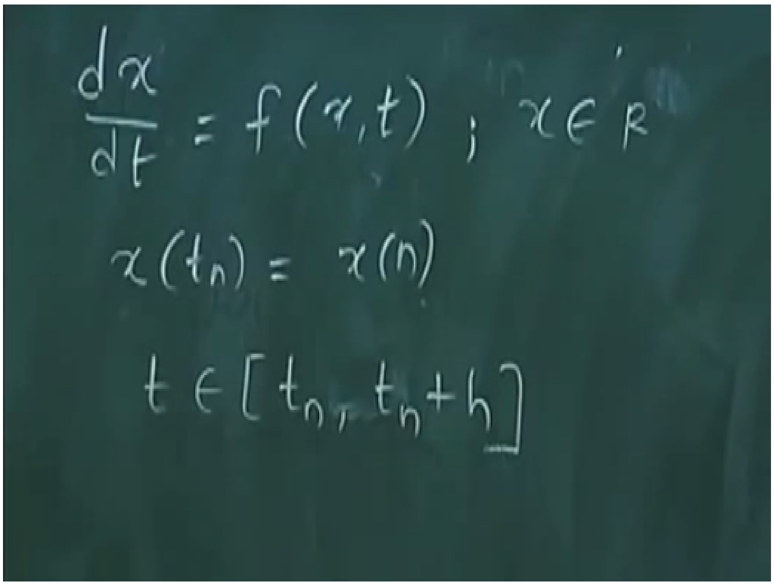
Under this class, I am going to talk about 2 types of method, 1 is multi-step methods. These multi-step methods are also known as predictor-corrector methods, some of the popular algorithm under this class is Gear's predictor-corrector. See you might have heard about Gear's integration algorithm. This in some book or some research paper you may have come across Gear's predictor-corrector.

So those Gear's predictor-corrector belongs to this class. The second method I want to see under this is orthogonal collocation. The orthogonal collocations method which we have seen or which we have used for solving or discretizing boundary value problems or discretizing the partial differential equations. This approach can also be used for solving ODE initial value problems and will have a peak at that too.

So we begin with multi-step methods or predictor-corrector methods. Now again my development is not going to be for multivariate case. My development is going to be for the scalar case. It is easier to understand okay and typically we do the same thing even for multi-step methods we do not derive coefficients separately for vector case. We just derive, make derivations for the scalar case and we use those coefficients in the vector case okay.

So same idea which was done for Runge-Kutta. Runge-Kutta is delight for the scalar case. We just use those coefficients okay for the vector case. So it is going to be the same case here.

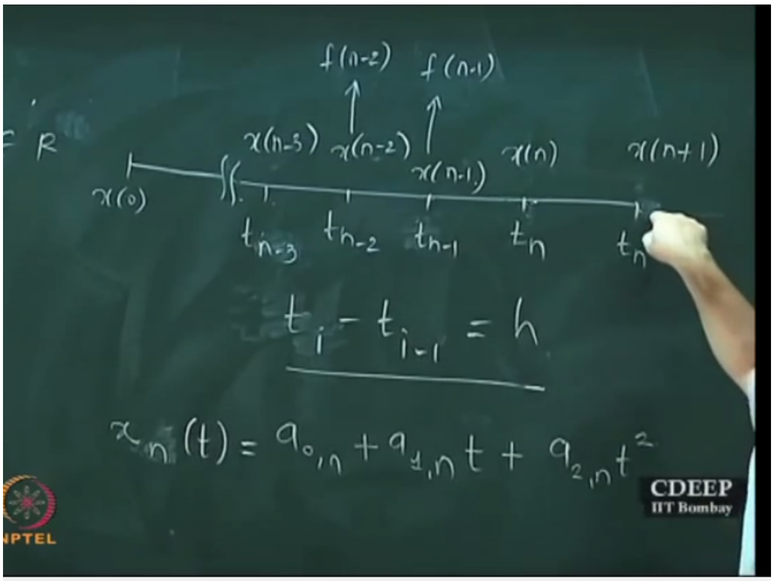
(Refer Slide Time: 21:58)



so for the derivation sake, I am going to be bothered about dx/dt=f x, t where x belongs to R, it is a scalar variable and we are at point x tn=xn and then I want to integrate over t so again I am going to solve the same problem, one small ODE initial value problem starting from time point tn and going to time point tn+1. Now when you say multi-step, let me clarify why this word multi-step is being used.

In Runge-Kutta methods, we were just worried about going from tn to tn+h okay. All that we used was xn and then values between xn and xn+1 we created some intermediate points okay and then we did evaluations at the intermediate points between tn to tn+h and then we did the calculations okay. Here the philosophy is different. Here what I am going to say is that well I have information available about what has happened in the past okay.

(Refer Slide Time: 23:48)



See I am currently at let us say I am currently at tn, I want to go to tn+h okay. So my problem is I have xn here and I want to find out xn+1. I want to go from here to here right but I have started my integrations somewhere here, this is my x0 okay and then I have this information about tn-1, tn-2, tn-3 right. I have information about xn-1, xn-2, xn-3 because I have started integrating from time 0 okay and I have this information.

When I have reached up to point tn, I have information in the past okay. I want to march from tn to tn+1 one step ahead in future. Currently I am in tn okay. I have this past information. Apart from this, I also have not only information of x at this discrete time points, I also have information about f n-1, I also have information about f n-2 right. These are values in the past. I can evaluate the derivatives at the point in the past okay.

I can evaluate I know the values of x or I know the values of the solution at time points in the past okay. What I am going to do is when I go from here to here okay I am going to make use of this past information okay. See it is like again if you take the analogy of you know when you are climbing down a mountain okay. When you take the next step, you may want to use information about what has happened in the past okay.

How was the slope and what is the local curvature, you want to use that information and make one step ahead okay. The Runge-Kutta method only you know looks at everything that is in the past is contend in xn and I just want to go from xn to xn+1 so Runge-Kutta method when it goes ahead, it is cautious, it takes small steps in between and goes okay whereas here we are not going to do intermediate calculations okay.

We are going to use calculations at these previous time points to make a judgment about how to go in the first. So 1 measure difference here is that these are fixed step size methods so tn or in general ti-ti-1 is h. So the step size is fixed okay. Now what is the philosophy? Philosophy is to use an interpolation polynomial. What is an interpolation polynomial? Interpolation polynomial is 1 which passes through given set of fixed points okay.

So one idea is that if I have this xn-1, xn-2, xn-3 okay, I can invoke Weierstrass theorem and say that well my solution x is actually a continuous function okay. A continuous function can be approximated by a polynomial. What kind of polynomial? I am going to fit a interpolation polynomial. So I am going to fit a polynomial using this data okay and do an extrapolation okay.

I fit a polynomial using past data and extrapolate from tn to tn+1 okay. So this will give me an explicit method. The other idea is I try to fit a polynomial using xn+1 so using future in the past okay then you get implicit formula because you will get the future is the function of you know this xn+1 is the function of xn+1 and you have to solve it relatively okay. So basically I am going to fit a polynomial.

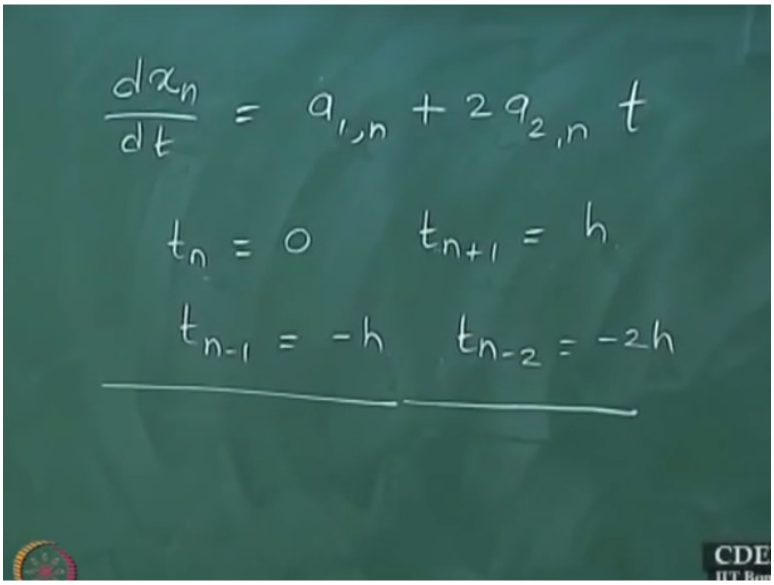
So my idea is to fit a polynomial. Now my notation is little bit complex okay and you have to carefully understand my notation. Let us say I want to fit a cubic polynomial okay or I want to fit a let us take the simplest polynomial quadratic polynomial okay. I want to fit a quadratic polynomial. So x0n+x1n t+x okay. This is the polynomial solution. Let us call this local solution xn t okay let us call this solution xn t.

What is this n? Okay this n corresponds to the point tn. I am standing at tn okay. I am not going to fit one polynomial okay. At this point, I am going to fit a polynomial locally, a quadratic polynomial okay and use it to do extrapolation. When I move on okay I am going to fit another polynomial, I am not happy with fitting one polynomial. It is not possible to fit one giant polynomial into all.

Suppose you are integrating from 0 to time 1000 minutes and you are integration interval is 10 seconds okay. I cannot fit a interpolation polynomial of higher order okay. I cannot fit one quadratic polynomial because the nature is changing as you move along the slope is changing. So I want to fit a local polynomial using local neighborhood data okay and then do an extrapolation okay.

So we are just going to use the local information and these polynomial coefficients are going to be time varying. They are going to change as you move in time okay. That is why this index sub index n comes okay. Now 0, 1, 2 is a quadratic polynomial, which I want to fit locally okay. This is fit in the sense I am not going to do least square fit here. I am going to do interpolation polynomial okay.

(Refer Slide Time: 31:48)

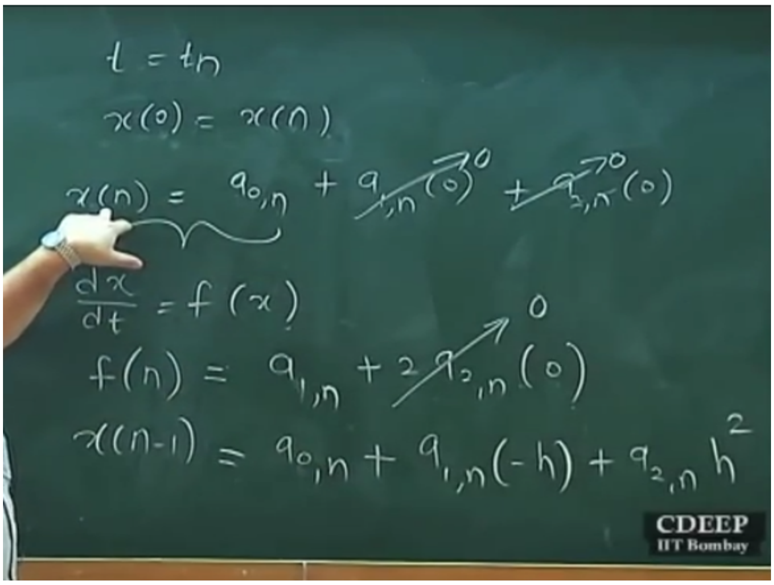


Let us see how we can fit this local interpolation polynomial okay. Now what I can do is I can differentiate this polynomial okay. What will I get if I differentiate the polynomial? a1, n+2 a2, n t okay. Now I am going to temporarily shift the time okay such that I am going to shift my time axis such that tn corresponds to 0 okay. For making this local fit I am going to shift the time axis such that tn corresponds to 0.

So what will be tn+1? This will be h. What will be tn-1? This will be –h. What will be tn-2? -2h and so on okay. Now when I am fitting the polynomial, in this case I am fitting a quadratic interpolation polynomial. How many coefficients are there? There are 3 coefficients, a0, a1, a2 okay. How many equations I need to exactly determine the 3? I need 3 equations so somehow I have to generate 3 equations okay.

So let us start doing this by with this shifted time scale okay I am going to now generate 3 equations and 3 unknowns and once I have solution of 3 equations and 3 unknowns, I have one way of doing calculations okay.

(Refer Slide Time: 34:07)



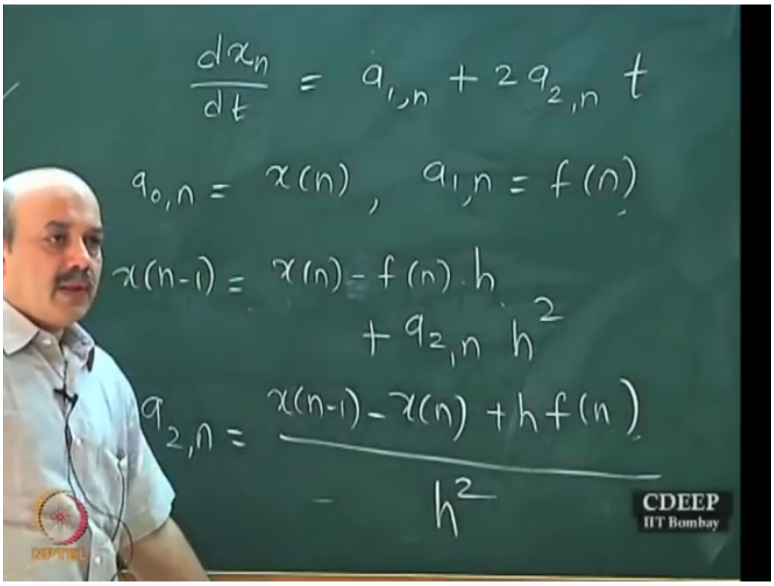
So now let us see how this is done okay. So what is x0 at t=tn we have shifted time now okay. In shifted time, what will be x0? Will corresponds to xn okay. Now xn in terms of shifted time, this will be a0, n+a1, n*0+a2, n*0 okay. I got the first coefficient. What is my first coefficient? Because this is 0, so this is 0, this is 0, I got x0n=xn okay. Now I need to generate 2 more equations okay. See dx/dt is nothing but f of x right.

Now what is fn? That is function evaluated at time tn okay. That will be=a1, n okay +2a2, n*0 fine. So this is 0 because with shifted time this is 0 okay. So I got second coefficient a1, n=fn okay. Now I want to generate the third equation. How do I generate the third equation? I can use the past okay. I can use information at xn-1 or I could use information of fn-1, I have a choice. I can choose from xn-1 or fn-1 okay.

So which one do you want to go? Xn-1 okay, we will derive both ways okay. So let us take initial possibility that to fit the interpolation polynomial, I am going to use xn-1 okay. So what will be xn-1? Xn-1 will be a0, n+a1, n*-h+a2, n h square right okay. Now can you eliminate and find out a2, n? Just tell me what is a2, n? Because a0, n is nothing but xn. We got this a0, n=xn.

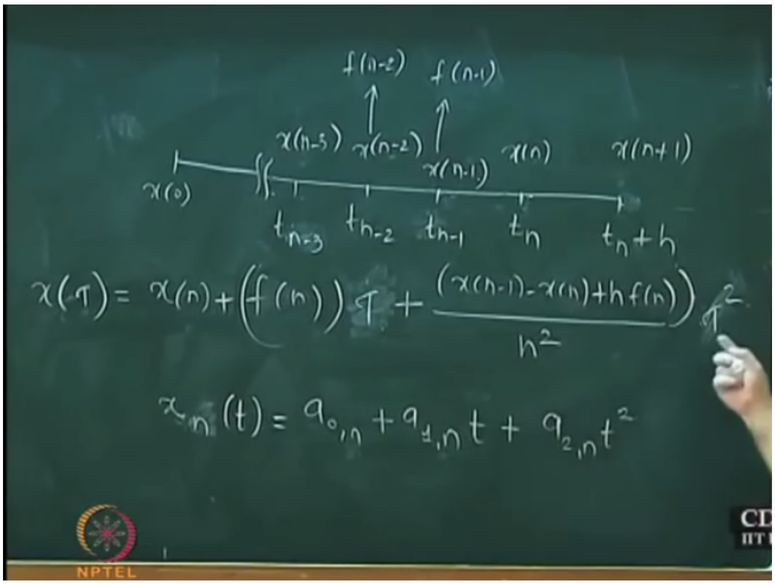
We got a1, n is nothing but fn so these 2 coefficients are known to us okay. What is the third coefficient?

(Refer Slide Time: 38:19)



So what I will get if I substitute there I know that a0, n=xn, I have found that a1, n=fn right. Now the third equation that I got is xn-1=xn-fn*h right+a2, n h square. So what is a2, n h square? So a2, n will be xn-1-xn+h fn/h square okay. This is my interpolation polynomial. This is my coefficients of interpolation polynomial. So I have found out interpolation polynomial with time varying coefficients. So what is my interpolation polynomial?

(Refer Slide Time: 39:42)

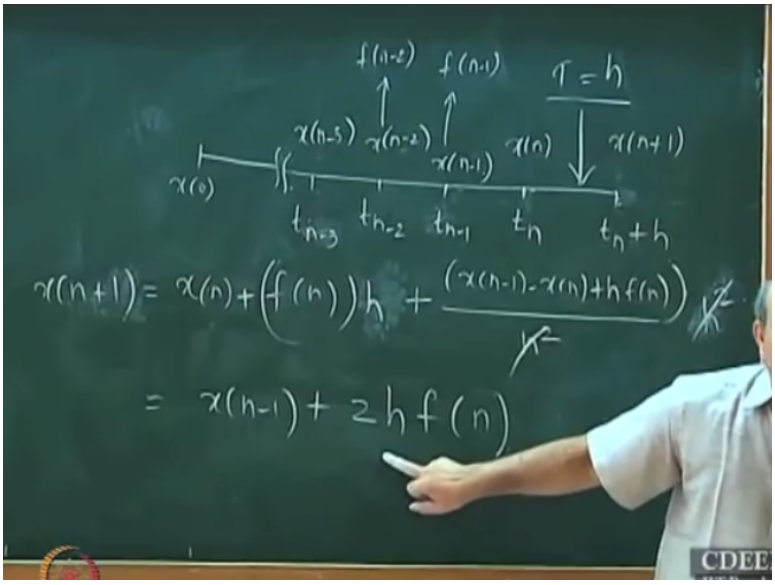


Let us go back and write it here. So my interpolation polynomial is now let us write it in terms of shifted time let us call it tau shifted time okay this is equal to what is a0, n is xn okay is xn+what is the second coefficient? Second coefficient is fn*tau, tau is the shifted time okay and what is the third coefficient? xn-1-xn+h fn/h square*tau square right. Is everyone with me on this? See I fitted an interpolation polynomial.

There are 3 coefficients a0, n a0, n turns out to be xn okay a1, n turns out to be fn okay and a2, n turns out to be xn-1-xn+h fn/h square*tau square this is my polynomial okay. This is the second order polynomial fitted using this point and this point in the past okay. Now I am going to do extrapolation. How will you do extrapolation? What I want to find out? The next step. So how will you get the next step? Tau=h okay.

Tau=h will give me this point so what will I get if I substitute tau=h here? Let us substitute. If you substitute tau=h, you will get xn+1, see setting tau=h will give you xn+1 okay.

(Refer Slide Time: 41:57)

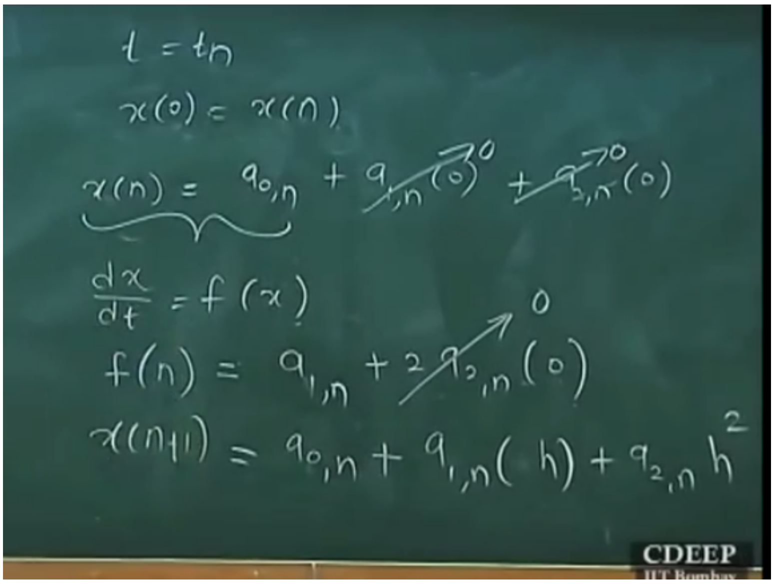


So if I set tau=h that is I want to go to this point then if I substitute that here tau=h I will get xh, xh is nothing but xn+1 okay, xn+1=xn*fn*h okay+here you will get h square, so this h square h square will cancel okay. Can you rearrange and write what you get now? Yeah so what you get? So after rearranging you will get xn-1+2h fn. Is that right? Okay. The final formula looks like this.

The final formula you do not see the interpolation polynomial anywhere okay. The interpolation polynomial, which we have fitted locally has disappeared with this final form actually what you have done is fitted a local polynomial. Is this the only way of doing this? No, okay. Let us do it some other way okay. What is this formula which you got? Is this an explicit formula or implicit formula? This is an explicit formula okay.

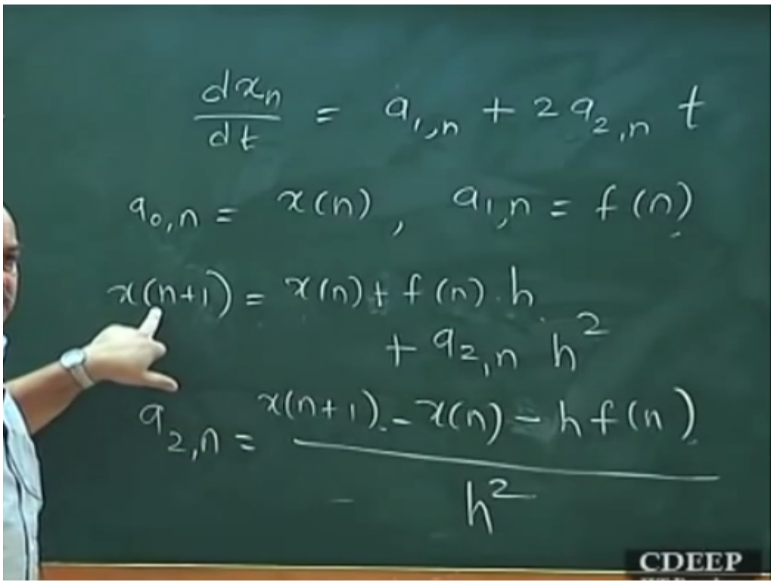
Because anything that is before is known to you okay. What is in the future is not known to you. Let us make a small modification and see what do you get now. This point I have chosen xn-1, instead of that let us choose here let us go back here okay.

(Refer Slide Time: 43:54)



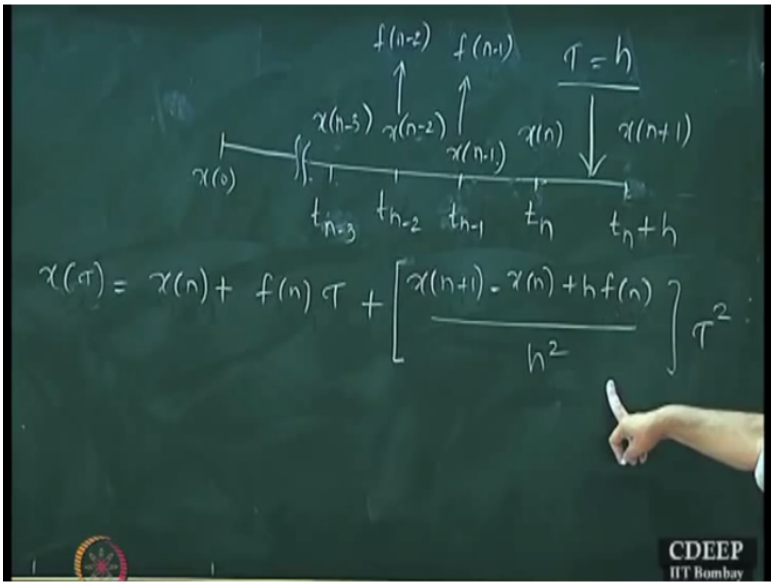
Make a small modification and I will choose xn+1, just derive what will happen. If I choose xn+1, this will be a0, n, this will be a1, n and this will be +h, -h will disappear and this will be h square. What will you get? Okay my equation here this a0, n is still xn, a1, n is still fn okay.

(Refer Slide Time: 44:25)



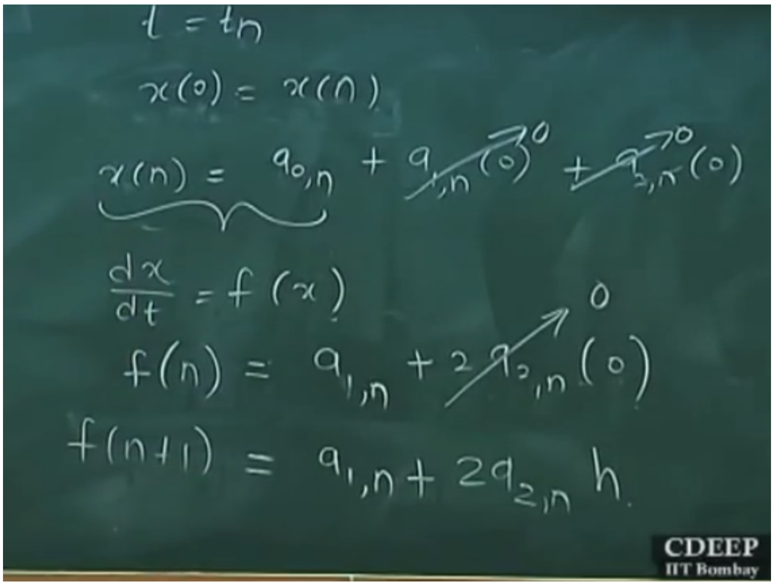
This equation will change, this will be xn+1=xn+h fn and this will be xn+1-h fn okay/h square that is your a2, n now okay. This has become xn+1 instead of xn-1, this is xn+1 xn h fn okay. Now what? How will the formula be modified? Just go back and derive the formula now. What do you get? You will still end up with the explicit formula I think. So what is the polynomial that you are fitting now?

(Refer Slide Time: 45:39)



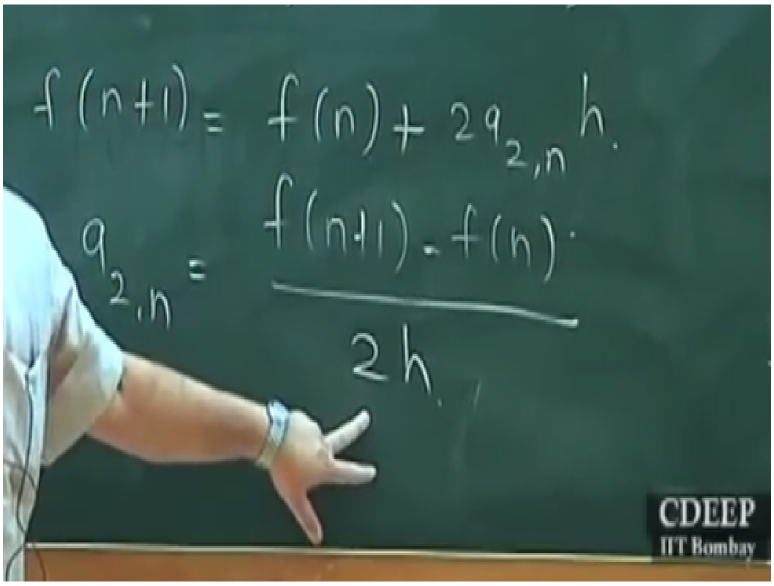
You will get x tau=xn+fn tau+what is the third coefficient? xn+1-xn+h fn/h square tau square okay. Now what do you get? You will get 0=0 here. Oh great so this will not help us. So what way to go? What about fn+1? Can we use fn+1? Just check. Now instead of using xn+1, I decide to use fn+1 okay. So here you get 0=0. So xn+1 is not useful. So we need fn+1.

(Refer Slide Time: 46:47)



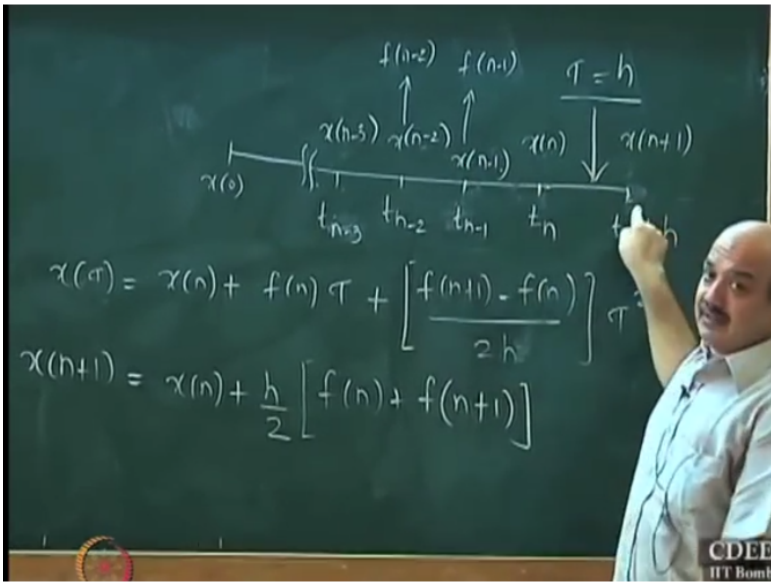
What is fn+1? fn+1 will be a1, n+2 a2, n*+h okay. Now using this I will get, these 2 coefficients remain same.

(Refer Slide Time: 47:19)



I will get fn+1=fn+2 a2, n h. So what is a2, n h? a2, n=fn+1=fn/2h okay. If you substitute this, what will you get? So my formula now becomes here.

(Refer Slide Time: 48:13)



This becomes so this is my time varying polynomial, this is fn+1-fn/2h*tau square okay. So if I set h=tau, so I will get xn+1. What is the integration formula? xn+1=xn+h/2 f of n+f of n+1 the famous trapezoidal rule. This is an implicit formula because xn+1 appears on the left hand side, fn+1 appears on the right hand side okay. This is our famous trapezoidal rule. I got an implicit formula okay.

When you see it in this form, you do not see the interpolation polynomial okay and what I want to stress here it is matter of choice how do you choose which point of it. See suppose I give you a problem in which you have to fit a cubic polynomial, what will you do? Cubic polynomial will have 4 coefficients, so you need 4 equations. How you generate 4 equations? There are variety of ways. You could do using xn, xn-1, xn-2, xn-3 okay.

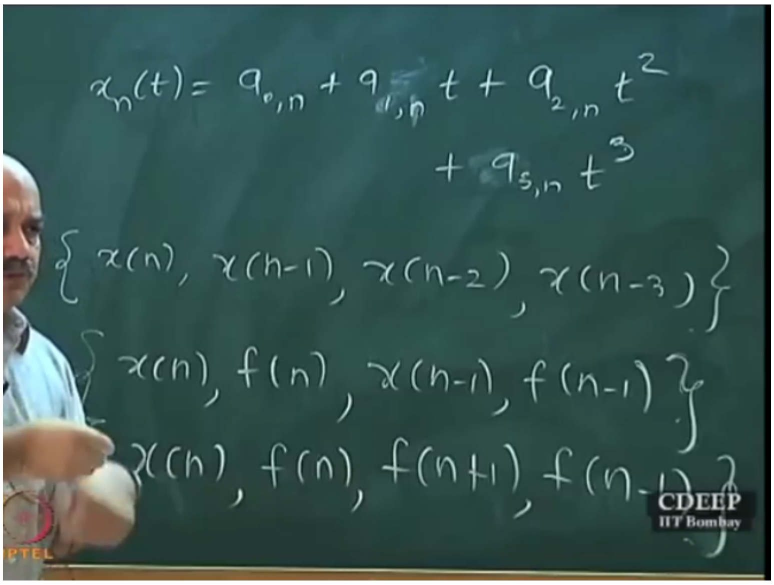
You could do using fn, fn-1, xn, xn-1, 4 equations okay. You could do using you know this and this okay, which points you use all of them are multi-step methods okay. So I will just summarize this here. **"Professor - student conversation starts."** Yeah you have a doubt. No, no, no. It will only give problem at a time 0. No, no, no. At time 0, you have to make some assumption about past yeah.

Why you started with the wrong guess, you start with the same initial point, same point in the beginning and then start creating new points, only first 3 points will be problem, 3 or 4 points. After that you have with the past right. No, there is nothing like a wrong guess know. This is not iterative solutions. See when you are solving ordinary differential equations, you are not guessing you are marching in time okay.

So do you know some values in the past is the question okay. So it is not just one initial condition, you need to know some values in the past. No, I can assume the same value was there in the past, it is same point in the beginning for last 4 instances. If I make the assumption, there is nothing wrong and I can go on marching. So after 3 instances, I will have past 3 and then I can go on doing it.

It does not get a problem so the convergence to the true solution depends upon something else okay. It does not matter even if you have slightly wrong initial 3 or 4 points okay. **"Professor - student conversation ends."**

(Refer Slide Time: 51:49)



So basically, if I want to fit a polynomial of the form you know xn t=a0, n+a1, n t+a2, n t square+a3, n t cube okay. I want to find out a0, a1, a2, a3 okay. To generate this, either I can use xn, xn-1, xn-2, xn-3, I could use xn, fn, xn-1, fn-1, I could use xn, fn, fn+1 fn-1. I want to create 4 equations in 4 unknowns, fit an interpolation polynomial okay of cubic type with time varying coefficients.

Those time varying coefficients can be either function of past 3 x values, they can be 2 derivative 2 x values, it can be 3 derivatives 1 x value, it is up to you okay. So every possible combination will give rise to 1 multi-step method okay. Final form the cubic polynomial or whatever that cubic form will disappear, you will only see an integration method with some coefficients okay.

What I will do in my next class is derive a generic mth order polynomial to be fitted in okay. I will derive a method formula for mth order method and so we can do any order fitting okay, find a multi-step method and derive it. So you should understand what is the philosophy, should be able to derive a new method if you want, that is very, very important, that is what I wanted to learn from this exercise.

Because the final rearranged form you know the origin disappears, it looks like some recipe you know, you take this, you take this, do this, do this, you will get the solution. What is the philosophy is not clear when you look at. So in the next lecture we will continue these multi-step methods. I will try to finish multi-step methods. There are different classes of multi-step methods.

One which only use past f1 will use only past x and so on and then there are variations, you get explicit methods, you get implicit methods. Then you use explicit method to initialize an implicit method. The same idea which we use for Euler you know, that is how do you get a good guess? Okay so you will get a good guess by Euler and then you know kick off your iterations.

And the same thing you do here, you take a nth order multi-step method, explicit and use it to initialize a nth order implicit method and that way you know you can converge faster. So large number of methods exists because the way you fit the polynomials is up to you. How much information in the past you want to consider relevant is up to you okay? So order you can choose how much data you want to choose in the past you can choose.

And then you can go on marching using interpolating polynomials okay. What is important to remember? There is no one interpolating polynomial, it is like series of interpolating polynomial so every time you move on, you are fitting a new interpolation polynomial, you move on you fit a new interpolation polynomial okay. So sequence of interpolation polynomials that is very, very important.