MATLAB Programming for Numerical Computation

Dr Niket Kaisare
Department of Chemical Engineering
Indian Institute of Technology, Madras

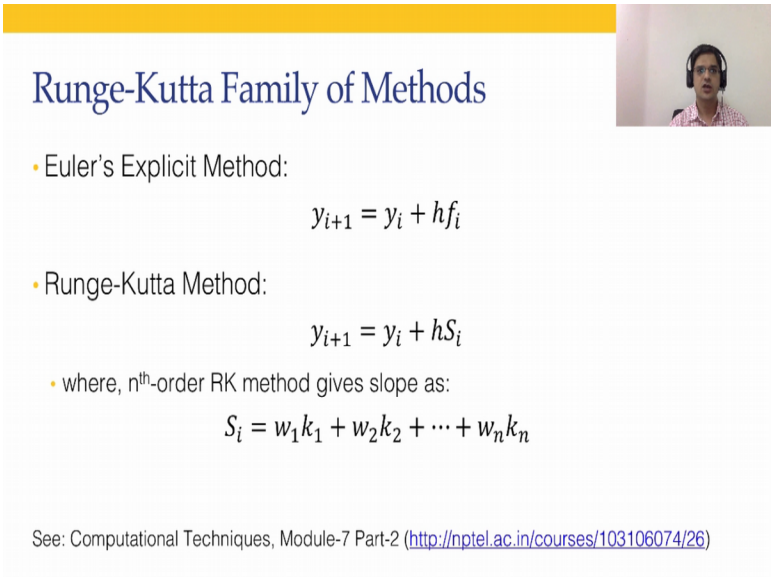
Module No. #07
Lecture No. #7.2
Ordinary differential equations – Initial value problems Runge Kutta (RK-2) Methods

Hello and welcome to MATLAB programming for numerical computations. We are in module 7. In this module, we are covering ordinary differential equations, initial value problems. In lecture 7.1, we introduced ourselves to ODE initial value problems and we used Euler's explicit and Euler's implicit.

We said in the previous lecture, at the end of the previous lecture, is that in the rest of this module, we are going to cover, explicit methods of the type known as Range-Kutta family of methods.

In order to introduce ourselves to RK methods, we will consider the second order Range-Kutta method. Before going to the, second order Range-Kutta method, I will give you a general idea of what Range-Kutta method looks like.

(Refer Slide Time: 01:06)

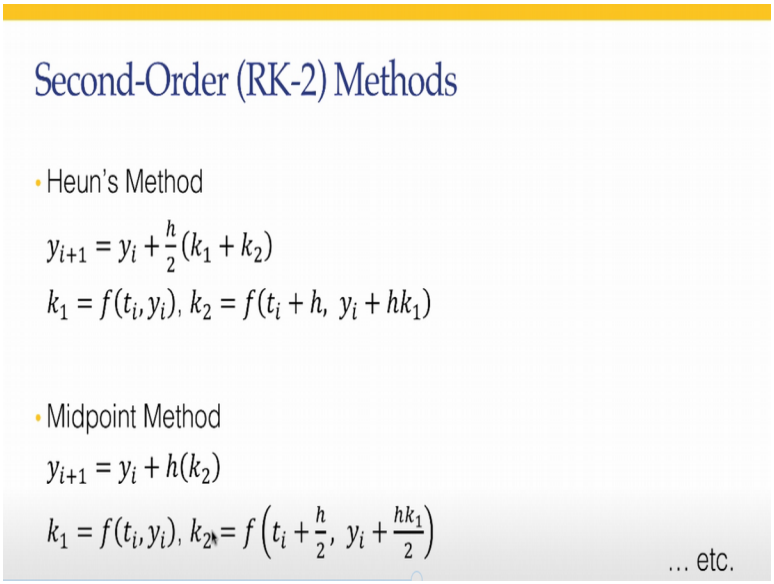


Euler's explicit method, we had Y (i+ 1) = Y (i) + h * f (ti, yi) okay, which we have written as fi. This is how we used the Euler's explicit. We also said in the previous lecture is that for getting higher accuracy, we can replace our fi, by some other way of calculating the slope Si.

And, we can use that slope, in the overall approximation Y (i+ 1) = Y (i) + h (Si). Now in nth order Range-Kutta method, gives the slope, as a weighted sum of, n different functions okay. Now in this case the nth, functions are going to be k1, k2 up to kn. k1 is going to be nothing but f (ti, yi). k2 will depend on (ti, yi) and k1. k3 will depend on (ti, yi) k1 and k2 and so on. kn will depend on (ti, yi) k1, k2 and so on up to kn-1.

Once we update the k1 to kn, we calculate the weighted sum, w1 k1+ w2 k2 and so on up to wn kn, substitute this over here. And we will get the next time solution using the Range-Kutta method okay. So, that is how the Range-Kutta method works. Range-Kutta method is a generic method for any nth, any number n. Most popular are, fourth order Range-Kutta methods. Range-Kutta methods the theory and discussions behind it, was discovered in module 7 part 2 of computational techniques course, the link for which is given over here okay.

(Refer Slide Time: 03:05)



Now let us consider, a couple of second order Range-Kutta methods okay. In Heun's method Y (i+1) is Y (i) + h /2 *(k1+ k2) which basically means our w1 and w2 are both 0.5 okay. k1 is nothing but, f computed at (ti, yi). And k2 is f (ti+ h, yi+ h) multiplied by k1 okay. So that is the formula for Heun's method. It is a second order Range-Kutta method because we have w1k1+ w2k2 only okay. That is why, it is a second order Range-Kutta method okay.

What we can do now, is we will go to MATLAB, we will start with the example that we took in the previous lecture and we will modify it to solve that using RK2 method. (Video Starts: 04:05) Okay this was, Euler's explicit method, let us save that as rk2Heun. Solve ODE-IVP using Heun's method, okay. This initial part remains the same, let us take tEnd as 5 and h as 0.1. Initialize the solution, in exactly the same way as before, solving using Heun's method okay.

So the first step, in Heun's method is going to be computation of k1, which is f (ti, yi) okay. So, our k1 = f (ti, yi). So, -2*T (i) *Y (i) that happens to be the same as fi, that we had over here, that is actually fine okay. tNew, let us call this particular guy as tNew. So tNew = T (i) + h, yNew is Y (i) + h*k1 okay. And next thing, we want to do is, calculate our k2. So, k2 is function of calculated at, -2 * tNew *yNew. And let us see Y (i+ 1) is Y (i) + h /2 * (k1+ k2) okay.

And that is our Y (i+1). Everything else remains the same, we do not need to change anything. So majorly the change that happened, was only in this part. Where, we changed from Euler's explicit method to Heun's method. Let us save this and hopefully when we run it, we will not see any more errors. We have run and now we have the solution, using the Heun's method. If you recall from lecture 7.1, the error, the maximum error was 0.033. Let us find out what the err is, in using Heun's method.

Max err is 0.002 instead of 0.03 that is 3 * 10 ^ -2. The error using Heun's method is, 2 * 10 ^ -3. So Heun's method, the RK 2 method, is more accurate than the Euler's okay. (Video Ends: 07:34) So, let us go on to Heun's method, and we will recap what we did. So, we started with the Euler's method code and we modified the Euler's method code as follows. First we did, within the for loop, we calculated k1 which is function of (ti, yi), k2 we calculated in 2 steps.

First, we calculated, tNew which is T (i) + h then we calculated yNew which is Y (i) + hk1. Then we calculated -2* tNew* yNew that was the function that we were interested. In midpoint method w1 is taken as zero and w2 is taken as 1. With this, this is the overall equation, k1 as

before is f (ti, yi), k2 as before is f computed at, sorry, k2 is different over here, I am sorry, k2 is f(ti+ h/2, Y(i) + hk1/2) okay. We calculated this substitute in k2 and we use the midpoint method.

(Video Starts: 08:40) Let us go again to MATLAB and solve this using midpoint method. This time I am going to solve it from scratch. Edit rk2Midpoint, solve. Copy the preamble part, and the initialization part, those we are known, going to remain the same. We will check it anyway, t0 is 0, y0 is 1, tEnd is 5, h is 0.1 and n is tEnd - t0 the whole thing divided by h. We have initialized our t vector, we have initialized our y vector and with y1=the initial condition okay.

Solve using RK2 midpoint method for i=1: n okay. So, first step is to calculate k1. k1 is f (ti, yi). So k1 = -2*t*y*T (i)*Y (i) okay. k2 is going to be -2*tNew*yNew okay. What is tNew? Our tNew is T(i)+ h / tNew ,yNew is yi+, yNew is Y(i)+ hk1 /2, Y(i)+ h * k1 / 2 and k2 is -2 tNew * yNew and Y(i+1) is Y(i)+ h multiplied by slope. And slope, that s slope is nothing but, k2 in this particular case. So h /k2, end plotting, plot (t, y) T. ^ 2 and err = abs (Y-Ytrue). Okay because we are using abs function it does not matter if it is (y-Ytrue) or (Ytrue-y) both mean the same thing okay. So we have this and let us save and let us run it. We hope that this runs without any error okay.

Let us, there might be an error okay, y new is undefined on line 20. So let us go okay. yNew, yes, the reason why I am getting this error is, I have defined this as small y, capital new, MATLAB is case sensitive, so this has to be small yNew. So, let us save this, okay again to emphasize, we need to be in order to debug a code, and we need to be able to understand what errors MATLAB is throwing.

MATLAB has thrown the error, undefined function or variable yNew on line number 20. That means, MATLAB does not understand or did not understand what the variable y, capital y, capital new, stood for. And the reason, why we got that error was because there was a type error. We changed yNew from capital y, capital new to small yNew to match yNew that we had over here okay. So, let us clear all and let us get to midpoint, okay now this time it runs without any error okay.

Let us now calculate the absolute error, value of absolute error, max err okay. And this is also, 10 ^ -3. So, the order of accuracy, of rk2 midpoint method, is similar to the order of accuracy of rk2 Huen's method. Both these methods are more accurate than the Euler's method.

Let us reduce, the h by 1 order of magnitude and see what we get. Let us save, h was changed to 0.01 and run and max error okay. The error has dropped from 10 ^ -3 to 10 ^ -5. So, the error has dropped by 2 orders of magnitude. Let us change this to 0.001 and let us run it, max error has fallen, by further 2 orders of magnitude. So, from this 1 of the things that might be clear to you guys, is that the order of accuracy for from global truncation error view point, for Euler's method was h to the power 1 whereas for RK method is h ^ 2.

We will consider the error formulation for RK methods, in a more formal way in a subsequent lecture. (Video Ends: 14:49) This brings us to the end of this lecture. In this lecture, what we have covered is introduce ourselves to, Range-Kutta family of methods and solve the ODE initial value problem, in single variable, using 2 different rk2 methods, rk2 Huen's method and rk2 midpoint method. There were two aims in here for us to do that.

One was to demonstrate that the rk2 method is more accurate compared to the Euler's method. The second thing was to demonstrate that, Range-Kutta is not a single method, but it refers to a family of methods. For any particular order, there are various different types of methods that are available, as well as there are different orders of accuracy. With this I come to the end of lecture 7.2. I will see you in the next lecture thank you.