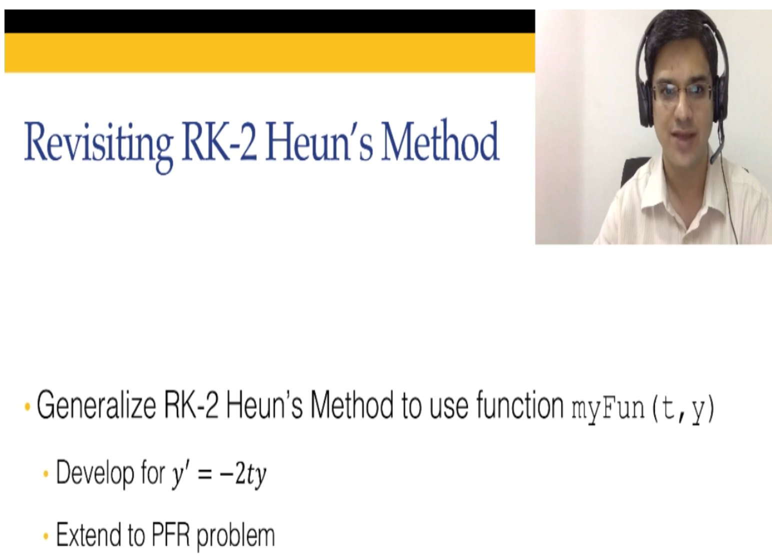
MATLAB Programming for Numerical Computation
Dr Niket Kaisare
Department of Chemical Engineering
Indian Institute of Technology, Madras

Module No. #07
Lecture No. #7.4
Ordinary differential equations – Initial value problems Runge Kutta Methods

Hello and welcome to MATLAB programming for numerical computations. We are in week number 7, in this module we are covering ordinary differential equations- initial value problems. So, far we have looked at, Euler's method and Runge-Kutta second order method and we have solved a couple of problems using them. We have also looked at ode45 algorithm that MATLAB provides in order to solve ode initial value problem.

(Refer Slide Time: 00:48)



In this lecture, we are going to go further and use higher order Runge-Kutta methods. Before, we do that, let us revisit RK-2 Heun's method. What we are going to do today, is the RK-2 Heun's method, that we had developed, for a specific problem y`= -2ty. We are going to modify that, to a generic problem that takes myFun (t, y) as an argument.

So myFun (t, y), basically generates, this f (t, y) and passes it on to RK-2 Heun's method. Let us go ahead and modify the code in order to do that. First we will do this, for the mod for the system y `= -2ty then we will extend it to the pfr problem. Let us go to MATLAB and do this okay.

(Video Starts: 01:38) So, this is the RK-2 Heun's method, that we had written in lecture 7 .2 okay. So, what we are going to do is, instead of this guy, be hard coded into this, what we are going to replace it with is, through using a function myFun okay.

So, let us first create a function myFun, function dy = myFun (t, y) and dy = -2*t*y okay. Let us save this as myFun okay. So, k1 should be replaced by myFun (t, y) okay. Right now k1, is basically going to be (ti, yi). So, I am just going to replace that (ti, yi) okay. So k2, is also nothing but f (tNew, yNew). The function f, is obtained through, again the same function myFun, so, we will just do this okay. And that should complete our problem.

Now, what we also want to do is, we probably do not need this additional (tNew, yNew) over here. So, I will just remove this and directly replace tNew with T (i) + h and I will replace yNew with Y (i) + hk1 okay. And this becomes, our RK2heuns method, which uses the function myFun. Let us save this and let us run okay. And it shows the overall curve, which looks exactly similar to what we had earlier okay. So let us, also look again at, max error, max error was 0 . 0021 similar to what we had gotten earlier in lecture 7 .2 okay (Video Ends: 04:43)

So, what we have done is, we have generalized RK-2 Heun's method to use function myFun. Now, what we are going to do is, to extend it to the PFR problem okay. (Video Starts: 04:56) So, runPFRode was the code that we had used earlier in lecture 7 .3, in order to run the pfrode. What I am going to do now is, in addition to solving using ode45. I will solve, using RK-2 Heun's method okay. So, for doing this, for all I am going to do, mostly, what I am going to need is this part and this part.

So, let us just take this. Copy and paste this over here okay. Prior to solving using Heun's method, we need to initialize. Initialization, is we just need to copy this part and paste it over here okay. So, our t, capital T is our independent variables, Y is our dependent variable. In this case Y has to be replace with concentration C, which are dependent variables okay. So, those are the only changes that we need to make okay. Let what, we shall do now is, just run that particular code okay. Let us run this code and we are going to start getting errors right from over here.

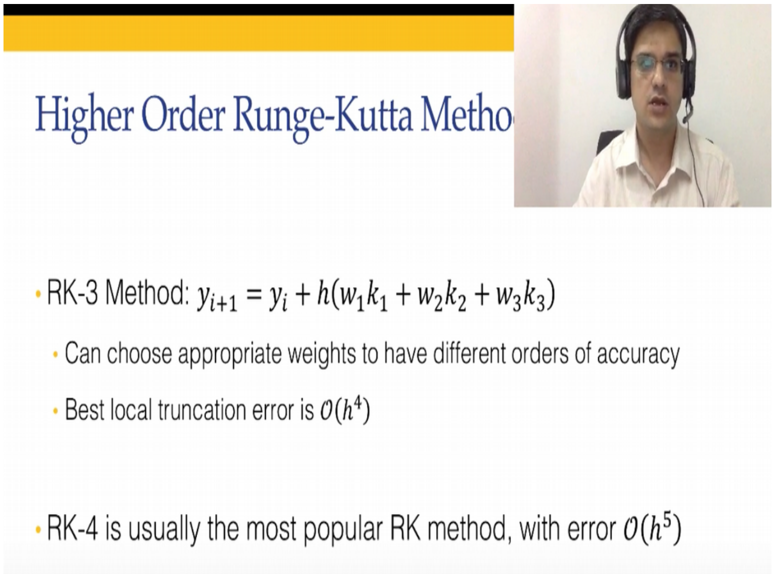
So, let us go ahead and do that, without worrying too much about errors. What we will do over here is, just change this end volume to V = 5 okay. And let us run this okay. We got an error, undefined function of variable t0. Keep in mind, that our independent variable was V or V0. So, that is what needs to be put over here. So, this will go from V0 up to Vend okay. That is our t in steps of h. So, let us also say, what h is going to be. The h is going to be 0.1. So, N as got to be vVend-V0, divided by h that is our n.

We have our y equal to zeroes (N+1, 1). And Y1 is the dependant variable in this case the dependent variable is C0. So I will just replace this okay. And I will say initialization, solving okay. And, what I we will do is plot. Plotting, results, plot (VSol, CSol) as blue line and (t, y) as dash red line okay. And the other thing that, we need to do is replace this with an appropriate function pfrFun okay. Let us clear all close all and run the runPFRode okay. And as you can see the red line, the red dashed line lies exactly or almost exactly on top of the blue solid line.

The red line is our Heun's method solver. Whereas, the blue line is our ode45 solver. So, as you can see over here, so this is the overall code that we have okay. Using the ode45 solver the code was very simple to obtain, using RK-2 Heun's method if you recall, this was the main code that we copy pasted from RK-2 Heun's over here, this was the main code.

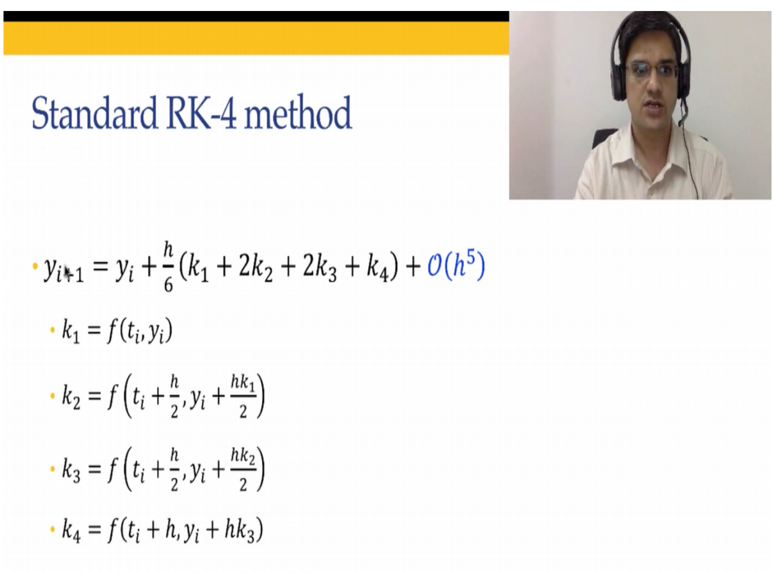
And in addition to that, we had to do some initialization, we had to initialize our h, we have to initialize our independent variable T, we have to initialize our dependent variable Y. Once we did that, we were able to solve this using RK-2 Heun's method okay. (Video Ends: 10:17) Okay, so what we have done is, revisited our RK-2 Heun's method and generalized it, to use a general function myFun and the pfr function pfrFun. Next, we are going to extend this to, RK-4 method.

(Refer Slide Time: 10:32)



The RK-4 method, is usually the most popular Runge-Kutta method. The RK-3 method uses three weights w1 k1+w2 K2+w3 k3 whereas; RK-4 method also uses a fourth weight + w4 k4. The RK-3 method has a local truncation error (h ^ 4). Whereas, RK-4 method has a local truncation error (h ^ 5). The best rk5 method also has an error (h ^ 5). As a result of this, we do not get a big advantage by using rk5 method instead of an RK-4 method. Therefore RK-4 method is arguably, the most popular rk method out there okay.

(Refer Slide Time: 11:32)



We will solve the first problem y`= -2ty using RK-4 method. The RK-4 method is given over here Y (i+1) = Y (i) + h / 6 * k1+2k2+2k3+k4 etcetera, etcetera okay. So, let us go to MATLAB (Video Starts: 11:45) let us open our RK-2 Heun's method. And save this as RK-4 standard okay. Solve ODE-IVP using RK-4 using Standard RK-4 method.

The initialization part remains the same. Initialize the problem definition remains the same. The initialization remains the same, solving using RK-4 method. k1 = this, k2 = myFun of something, k3 = myFun of something, k4 = myFun of something and Y (i+1) = Y (i) +h /6 * k1. Let us go here and see k1+2k2+2k3+k4. k1+2*K2+2*k3+k4. Our k2 was function (T (i) + h/2, Y (i) + h*k1/ 2) okay.

I will just copy this and paste it in k3, and we will do new modification as required. k3 is T(i) +h/2 again but Y(i)+h*k2/2. Instead of k1 /2, we now have k2 /2 okay. So, let us paste and the only thing that needs to change, this k1 is to be changed to k2 okay. Same thing, we will paste for k4. And let us see what change needs to be done. k1 as k4 sorry, is a function (T(i)+h, Y(i) +hk3). So, let us paste it T (i) + h and not T (i) + h/2. And Y (i) + h*k3 and not divided by 2.

So, we will delete, change k1 to k3 and remove divided by 2. And this is what Y (i). Let us save this and let us first clear everything and let us run our RK-4 standard method okay. And we see the, same curve and the curve looks similar to what we had earlier. Let us check on errors or let us check on max error, the max error is 7.5*10 ^ -6 as we see. When we went to RK-4 standard method, there was significant drop in the error. With the error drop from around 2*10 ^ -3 to 7 into 10 ^ -6 okay. (Video Ends: 15:27)

So, for the same step size, RK-4 method is significantly more accurate than RK-2. And that is one of the reasons for the popularity of RK-4 method okay. So with that, we come to the end of this lecture. What we covered in this lecture were two things. First we improved our RK-2 code. So, that we are able to use a generic function myFun (e, y) instead of hard coding it in inside the code itself. Next, we extended our RK-2 Heun's method to RK-4 standard method. So with that I come to the end of lecture 7.4, and will see you in the next lecture. Thanks and bye.