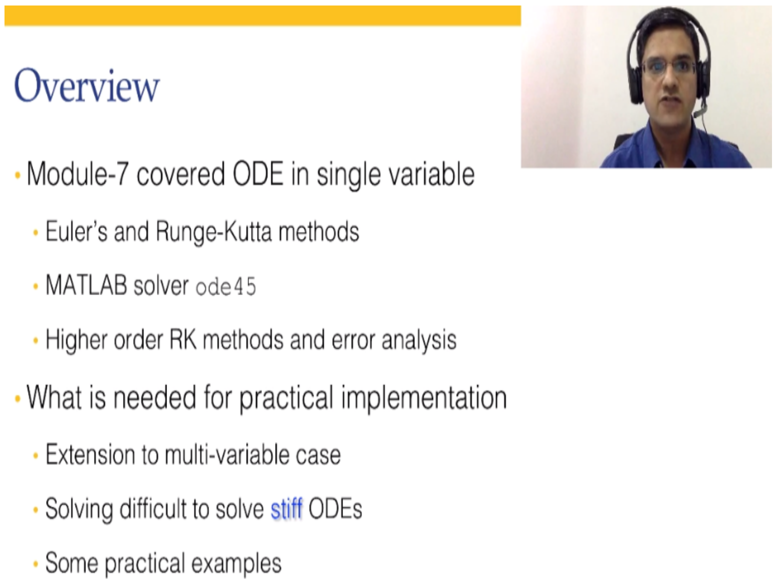
MATLAB Programming for Numerical Computation

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

Module No. #08
Lecture No. #8.1
Practical aspects of ODEs – Multi variable ODE

Hello and welcome to MATLAB programming for numerical computations. We are in the last week of this particular course. In this week we are covering module 8 practical aspects of solving ordinary differential equations.

(Refer Slide Time: 00:26)



Overview

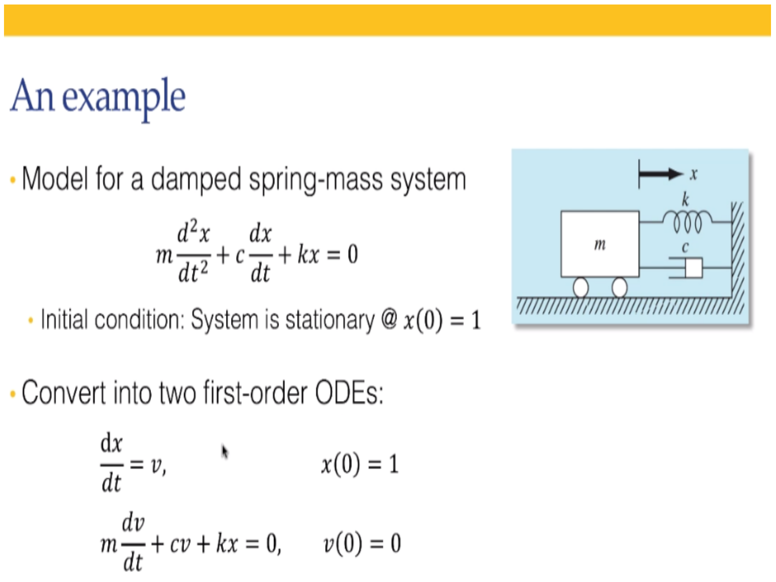
- Module-7 covered ODE in single variable
  - Euler's and Runge-Kutta methods
  - MATLAB solver `ode45`
  - Higher order RK methods and error analysis
- What is needed for practical implementation
  - Extension to multi-variable case
  - Solving difficult to solve stiff ODEs
  - Some practical examples

So what have we covered in module 7 is listed over here. We started with Euler's method both implicit and explicit method. We showed that Euler's explicit method is going to be stable only for a small range of step size. Thereafter we talked about Runge-Kutta methods. Then we talked about MATLAB solver ODE45. However we considered problems in single variable only.

Finally in the last 2 lectures of that module we covered higher order Runge-Kutta methods and error analysis. With higher order Runge-Kutta method we specifically covered the RK-4 method. In this module we are going to cover extension to multivariable case. We are going to cover how to solve difficult stiff ODEs before we go on how to solve the stiff ODEs we will cover what stiff

ODEs actually mean. And finally we will finish off with some practical examples in the last 2 lectures of this module.

(Refer Slide Time: 01:27)

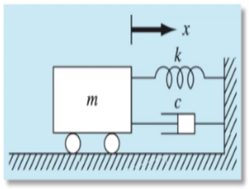


An example

- Model for a damped spring-mass system

$$m\frac{d^2x}{dt^2}+c\frac{dx}{dt}+kx=0$$

  - Initial condition: System is stationary @ $x(0)=1$



- Convert into two first-order ODEs:

$$\frac{dx}{dt}=v, \qquad x(0)=1$$

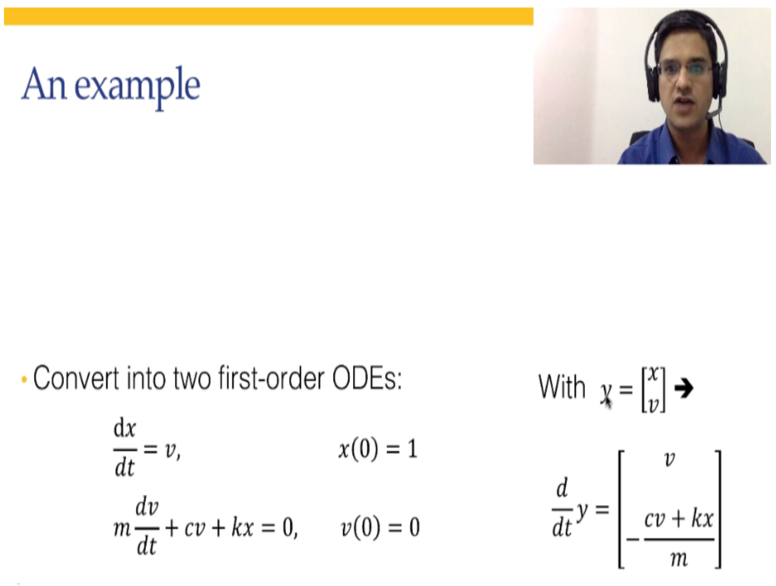
$$m\frac{dv}{dt}+cv+kx=0, \qquad v(0)=0$$

So let us get started with an example. This is an example, a textbook example for numerical methods courses. The overall equation as you know is given by this. This is a mass and we displace this mass by a certain amount and there is a spring and there is a damper and because of this displacement this particular mass is going to oscillate.

What we are going to consider is this particular model, the initial condition that is required is as stated over here. At time t = 0, we have displace this mass by a distance of 1, the velocity is 0 and we release the mass. So this as you see is a second order ordinary differential equation. What we need to do is to convert this into a system of first order ordinary differential equations. The way we do this is we realize that dx /dt can be represented by another variable v and we can write this as m dv /dt + cv + kx = 0.

So when we write that we are going to get 2 first order ODEs. The first equation rather is dx /dt = v. The initial condition is at t=0 x is 1. At t=0, v is=0. We are going to convert this into matrix differential equation and that is given over here.

(Refer Slide Time: 03:02)



An example

- Convert into two first-order ODEs:

$$\frac{dx}{dt} = v, \qquad x(0) = 1$$

$$m\frac{dv}{dt} + cv + kx = 0, \qquad v(0) = 0$$

With $y = \begin{bmatrix} x \\ v \end{bmatrix} \rightarrow$

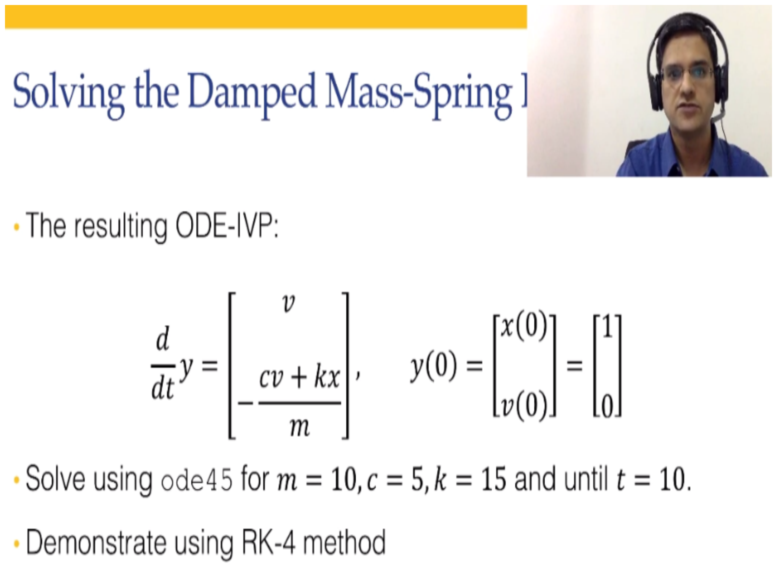
$$\frac{d}{dt}y = \begin{bmatrix} v \\ -\frac{cv + kx}{m} \end{bmatrix}$$

We are going to define our vector y as x v. As we have been doing throughout this course, we are going to define all our vectors as column vectors. That means we are going to have n rows and a single column. In this case there are 2 variables and therefore we have 2 rows and a single column and we have y is defined as x and v and dy / dt is the first guy is v. The second guy is –cv + kx whole thing divided by m. So let us go on to MATLAB and try to solve this problem.

(Video Starts: 03:44) Okay, so the first thing that we are going to do is to create a function that will calculate dy for given values of y and t. springFun okay and as we have been doing so far. function fval=massSpringFun (t, y) okay. If you recall, this line remains exactly the same. Function for mass spring system using ODE45 okay. So what is the first thing that we are going to do is we are going to extract our x and v.

So our x = y1, v = y2. Calculating our fval. fval1=v. And fval2, fval2 is going to be negative of (cv + kx) / m. It is negative c * v + k * x / m okay. The other thing that we are going to do is we need to define our fval as a column vector. So rather than writing f1=v and f 2=this right hand inside, we are going to write (f2, 1) is this and (f1, 1) is this. We have not yet defined our constant c, k and m okay. So let us go and do that. Define constants c = k = m =. Define dy / dt okay.

(Refer Slide Time: 06:02)



Solving the Damped Mass-Spring

- The resulting ODE-IVP:

$$\frac{d}{dt}y = \begin{bmatrix} v \\ -\frac{cv+kx}{m} \end{bmatrix}, \qquad y(0) = \begin{bmatrix} x(0) \\ v(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

- Solve using `ode45` for $m = 10, c = 5, k = 15$ and until $t = 10$.
- Demonstrate using RK-4 method

So let us go to MATLAB and sorry, let us go to power point and see what values we have. So this is what we are going to solve it for m=10, c=5, k=15. So m=10, k=15, and c=5. K=15 and c =5. So we will just save this okay. So now let us make a driver function that runs this mass spring damper system. So let us call this, edit solveMassSpring and it will create a new script. To run the mass spring system okay.

And what do we have over here we have we need to give the initial conditions. The initial conditions is y0=1,0. y 0=1;0 because that is our column vector. And we want to run this from t=0 to t=10. So as we have done in the past t= (0 10) okay. And then what we need to do is we need to use ODE45. So how do we solve using ODE45 tSolution, ySolution= ODE45 @ (t, y).

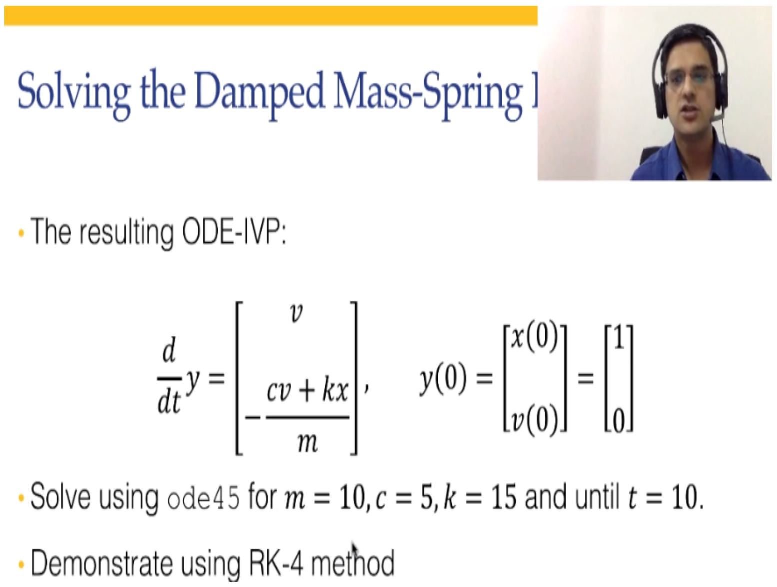
Remember this is going to be the syntax that we are always going to use for ODE45. (t, y) space the name of the file that is massSpringFun. MassSpringFun (t, y) okay. We want to run this. Let us call this as tSpan instead of t. To make it very clear that it is a span of times. (tSpan, y0) okay. And finally we need plot. So plot (tSol, ySol). We actually need to solve to sorry, we actually need to plot the locations versus time.

So the location is in all of the rows and it is going to be in the first column okay. And let me comment solve using ODE45.Plot the results. Let us run this okay so we have run this and this is an under damped mass spring damper system. We start with location at 1 and it oscillates before it will finally go to steady state. Instead of 5, let us save it increase it to 50 and let us run the system and see what we are going to get okay.

As you can as we increased the damping coefficient we do not get oscillatory response but we get a response like this. What we will do next is we decrease the c instead of 5 or 50. We will decrease it to 1 and see how the response is going to be. As you all might remember from your physics classes. The response is going to become more oscillatory. So let us run this and see. As you can see this compared to what we got at c=50 this is significantly more oscillatory.

So what we have done is the overall ODE45 the way it works for a single variable and multiple variables is nearly exactly the same. You are you need to define the function exactly as we did in single variable case. Name of the functions (t, y). t is going to be scalar as it was before. However y is a vector. Keep in mind that y is going to be a column vector. We want this to be column vector just for uniformity.

And it should return fval. In the previous module we have returned fval as a single value. In this particular multivariable case we are going to written fval as a 2/1 vector. The size of fval has to be the same as the size of our vector y. Why because fval returns dy /dt. What we are going to do next is to what we, what to do if we have RK-4 method not single variable RK-4 but if we had to do a multivariate RK-4.
(Refer Slide Time: 10:47)



Solving the Damped Mass-Spring

- The resulting ODE-IVP:

$$\frac{d}{dt}y = \begin{bmatrix} v \\ -\frac{cv+kx}{m} \end{bmatrix}, \quad y(0) = \begin{bmatrix} x(0) \\ v(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

- Solve using `ode45` for $m = 10, c = 5, k = 15$ and until $t = 10$.
- Demonstrate using RK-4 method

Until this point, what I have covered is something that I want you to play pay close attention to. Because that is something that you are going to use in multiple real scenarios. RK-4 method that I am going to show in the next few minutes is just for demonstration purpose. It is just for you to see how everything in Runge-Kutta method or in most of the numerical techniques follows similar rules for multivariable case just the way we had for a single variable case.

The only difference is now instead of a single variable you have to be just a little bit careful in tracking a multivariate case okay. And edit my RK-4 okay. So this was the solver that we had generated in our previous module okay. So let us go ahead and change this power y0 is (1; 0). So let us make that change y0 is (1; 0). Our t0 was 0 and tEnd is 10. So tEnd, let us change that from 5 to 10.

So this is our t0, this is our tEnd, this is our y0 okay. Let us keep our h=0.1 as before and our n as before is going to be this. Now when we are to initialize the solutions, our time vector is going to be just a single vector as we have over here okay. What we are going to do with y is that y, we are going to denote this as a 2 by n+1 vector.

Rather y at times 0 will be in the first column, y at times 0.1 will be in the second column y, at 0.2 will be in the third column so on and so forth. So that is the reason why we are initializing y in this manner okay. Next thing we are going to do is populate the initial value of y and we are going to do that with (y:, 1).

What that means is all the rows in the first column are going to be populated by our initial condition (1, 0) okay. Now solving using RK-4 method but we need yi. So yi we will say = yi okay. So k1 is going to be equal to myFun now the name of the function has changed to massSpringFun okay. So let me just copy this and paste it everywhere okay. So what I have done over here is, I just said Yi=yi okay and then replaced all of Yi's with yi okay.

There is going to be an error with this and I will come to that in a minute but let us write this as it is okay. And our y i+1 we will just leave it as this okay. So let us keep this and let us run and try to see what is going to happen. This is going to give us an error and that is because I have not kept track of the fact that yi's are now vectors. So let us run okay what we get is that index exceeds matrix dimension and that is because over here.

We have taken our yi as scalar. If we type yi, yi is 1, what I should have been doing is to give the first column of the capital y vector sorry, capital y matrix. So the first column is going to be :, i when i=1 okay. So as you can see the thing that has changed from RK-2 sorry, from RK-4 single variable to RK-4 multivariable is that instead of taking a scalar value, now I have to take that entire column. That is 1 thing that has changed nothing really has changed significantly over here.

The other thing that I will change is this, I will write this as yNew. I will write this as yNew=Y (i) +h/6 into weighted some of case. And then I will write y all of the rows in the i+1th column will be equal to yNew. And that is going to be what I am going to do and Yend that is y at the last point is going to be the last column. Last column is all the rows in the end column and that is going to be :, end that is going to be our y end and plot (t, y) okay.

So let us save this and run it and hopefully this will run without any error okay. And this is how our solution looks like with RK-4 method. I have plotted x as well as the velocity. Let me just plot x over here by saying the first row and all the columns. The first row has the location; the second row has the velocity. So if I have to plot only the location, I am sorry, if I have to plot only the locations then I will have to give this command.

And let me run this okay and this is the result that we get using RK-4. Let me click over here and say hold on. So that this plot is held. And I will solve this using ODE45 and compare the results by saying plot dashed red color line. So let us save this and let us run this let us look at the plot. As you can see the solution using ODE45 falls exactly on top of the solution using RK-4.

And the reason it does that is RK-4 is a higher order method and its accuracy is similar to that you will see in ODE45. (Video Ends: 17:50) Okay with that I come to the end of this particular lecture. What I have covered primarily in this lecture is to extend how you will solve ODEs from single variable to ODEs in multiple variable using ODE45. Thereafter I covered something that is beyond the scope of this particular course.

What I covered is how you can simply extend single variable RK-4 method to a multiple variable RK-4 method. But the take home message primary message from lecture 8.1 is if you want to use MATLAB solvers ODE45 for multivariable case going from a single variable to multiple variable case is fairly straightforward. With that I come to end of this lecture and I will see you in the next lecture thanks and bye.