Advanced Aircraft Control Systems With MATLAB / Simulink

Prof. Prabhjeet Singh

Department of Aerospace Engineering

Indian Institute of Technology Kanpur

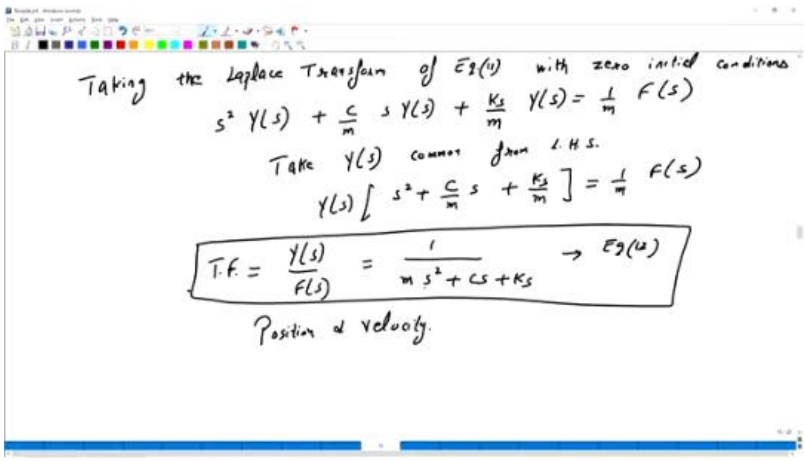
Lecture 52

MATLAB simulation of mass spring damper system

Hello friends, welcome again. In the last lecture, we discussed second-order systems, specifically the mass-spring-damper system. Let us simulate that system in MATLAB using different methodologies. So, I have considered a few values here. Time, I would consider from 0 to 4 seconds with a frequency of 0.01. Mass is 18 kg. Stiffness, K, 70. I'm converting it into newtons by multiplying by 9.81. Similarly, damping, 6 multiplied by 9.81. And the force, I'm assuming it to be 5 kg. So in newtons, it will be 5 multiplied by 9.81. And the system, I am representing it as a transfer function. So, TF stands for transfer function. Now, let me write here: help tf for the syntax of the transfer function. So, tf numerator comma denominator.

So in numerator we have 1. If you see the equation here for the transfer function that we had derived earlier. So in numerator we have 1 and in denominator we have m into s square plus cs plus ks. So here the coefficient for s square is m similarly coefficient for s is c and similarly for coefficient of s to the power 0 is ks that we have represented here m comma c comma k m comma c comma k I am representing here it as k instead of ks. Then I am defining the input u equals to f into 1s of size of t. Now if I let me just remove the semicolon from here and run this.

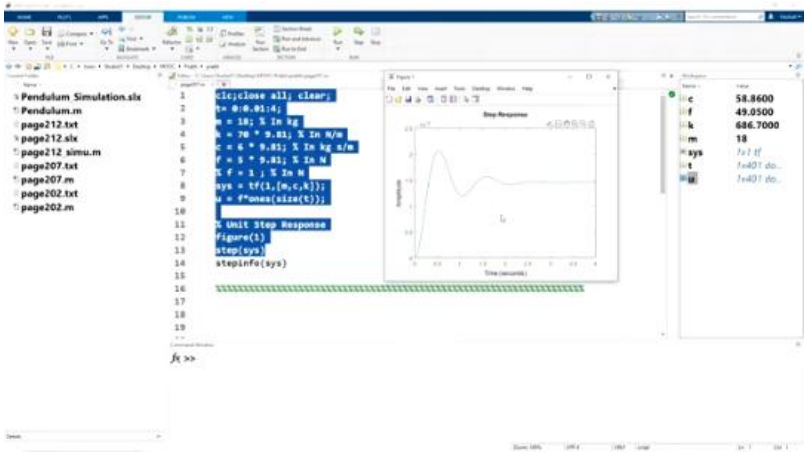
(Refer Slide Time: 01:42)



So here the system is represented in a transfer function form. Here 1 upon 18 s square plus 58.86 s plus 686.7. This is a continuous time transfer function. And u is again now here as a vector which is nothing but f into 1s of size of t. So what is size of t here? You can see here from the workspace it is 401. So I am multiplying that vector with 1 that is 1s. Into F here, F is nothing but five into nine point eight one. Now, the first step that we do here is for plotting this unit unit step response. Let me just clear it. I'm representing it as figure one step.

Step of S Y S. Let me just run this code. So this is what the amplitude of a mass spring damper system is, the displacement. It is going up to 1.5 into 10 to the power minus three in about four seconds. So here input, this is a unit step input. And if you want to see the parameters here, what is the peak time?

(Refer Slide Time: 03:34)



You can represent, you can find that by writing the syntax here, step info SYS. Here you can see we have rise time, transient time, settling times, maximum settling time and peak

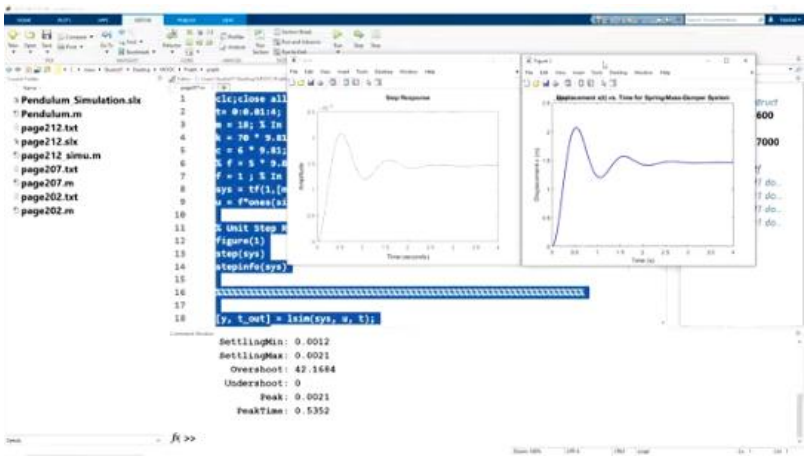
time. You can also view in the figure. Here the peak response is about 0.535 seconds. Here we have 0.535 seconds. Similarly, rise time. Rise time is 0.208 seconds. 0.2078 this approximately 0.208 seconds similarly we have settling time settling time is 2.27 seconds so this is how we can plot a unit step response by using step command again if you want to explore this You can simply write here help and sys and you can have the information readily available here. Now let us proceed further.

Now instead of unit step response, I want to see the simulated response. with different control input that is it will not be a unit step response so this can be represented is can be simulated using the command lsim, lsim we have already if you have been practicing MATLAB simulink example before you must be aware of this command lsim, else let me give you a brief overview of this lsim command So it computes the time response simulation data of dynamic system to arbitrary inputs. So this MATLAB function returns the system response y to the input u sampled at the same times t as the input. So the syntax is lsim that is system then followed by input then followed by time.

So the system is represented I have represented it as sys that is what we can see here sys input is u I mentioned input here u which is a vector and time and the output will be your states that is position and velocity and the time t out so we will get y and t as the output vectors and this i am representing this in in a figure two so plot t comma t underscore out comma y comma b b represents The color of the lines and the line width I have given it as 1.5. Title I mentioned as displacement versus time for spring mass damper system. X axis represents time and Y of course represents the displacements X or in meters. Now let me run this code. Now see here we almost get a similar response here. Let me uncheck these variables. Now here we observe we get almost a similar response here. Now one thing here students get confused about why there is a difference in magnitude in the responses of the amplitude that is the displacement. Well, the reason is that the input is not a unit step input in this case. Now, if I give a unit step input, let me uncomment this. I am pressing Control+T. I am commenting this line. Now, here the input is 1. Now, if I run this code.

So now we can see that even the magnitudes are matching. So that means we are getting the same amplitude or the displacement x in meters. Now, after this, how do we plot the velocity? So there are multiple ways. So let us try one of the methods to find the velocity.

(Refer Slide Time: 08:07)



We multiply the displacement transfer function by S. S is nothing but the Laplace variable, which corresponds to differentiation. In the Laplace domain. Thus, the transfer function for velocity becomes So what we do is multiply S with the displacement transfer function. What is the displacement transfer function? This one:

$$G_v(S) = S.\frac{1}{mS^2 + cS + K} = \frac{S}{mS^2 + cS + K}$$

Now, let us switch to MATLAB again. Now, see here the transfer function for velocity. So, here the numerator is now 1 comma 0. So, here in the numerator, we have The coefficient of s is 1, and s to the power 0 is 0. So, the coefficients for the numerator are 1 and 0, which in vector form we can represent in these closed brackets. And the denominator is again m, c, and k, which remains the same as the denominator of the transfer function. Now, the system transfer function again, we have to represent it by using the command, the syntax tf. I have changed the variable name to sys underscore v equals to tf numerator comma denominator. Then, finally, we do the same process. We follow the same process here to simulate the velocity response. Here, the output I have represented as velocity and the tout.

Output vectors are velocity and t. And I have again used the lsim syntax here. sys_underscore_v, which is this variable. The input u, comma, time variable t, and finally, I am plotting. I am naming that figure as figure 3, plot x, comma, y. In the x variable, we have t, which is this t_out, and in y, we have velocity. b again represents the color of the line, and the line width I am considering as 1.5 so that it is clearly visible. The title is 'Velocity versus Time for Spring Mass Damper System.' X-label: on the x-axis, we have time, and on the y-axis, we have velocity.

Now, let me run this code. Now, see here. Now, we have that velocity is actually approaching zero. So, this is how you plot a velocity curve for a mass-spring-damper system. In fact, for any other system, we can use it. Now, there is another method. We can obtain the velocity. Of course, the velocity is represented as

$$V = \frac{dx}{dt}$$

Now, what is x? x we have actually found here. In our previous lectures, let me just go back to MATLAB. What we'll do is we'll differentiate that equation directly in MATLAB. So, x is

$$y(t) = fK\left[1 - \frac{\omega_n}{\omega_d} e^{-\xi\omega_n t} \sin(\omega_d t + \phi)\right]$$

This was equation number 9. So if you notice here, I have written dot here because here time is actually a vector. So we need an element to element multiplication. That is why I have written here dot. Otherwise MATLAB will simply throw an error. And what is ω_n and ω_d?

$$\omega_n = \sqrt{\frac{k_s}{m}}$$

$$\xi = \frac{c}{2\sqrt{k_s m}}$$

$$\omega_d = \omega_n\sqrt{1 - \xi^2}$$

$$\phi = tan^{-1}\frac{\omega_d}{\xi\omega_n} = \tan^{-1}\left(\frac{\sqrt{1-\xi^2}}{\xi}\right)$$

That is why we were going in detail about these expressions in the earlier lectures. I have represented here k_s by k and then simply I have written the expression with the variable name x underscore t that equation number 9, Now what we need to do is we need to differentiate this equation with respect to time. So the syntax for the differentiating is DIFF. You can search for this command help DIFF.

So the difference is an approximate derivative. This MATLAB function calculates differences between the adjacent elements of X. So there are different syntaxes for this. I have used dx, DIFF, X of t divided by dt. Dt is the time interval. So here dt I have represented it as second vector minus second element of that vector minus the first

element. So $t(2) - t(1)$. Now, if you follow this way, then the time vector will actually be one element less than the that we have used before. The previously time element is 401. If I run this code and time velocity, this element will be 400. So, the way to represent, way to write it as t of 1 is to end minus 1.

So, this time vector for velocity is adjusted for differentiation. Then finally, plotting the velocity response, that is Figure 4: plot t_velocity (the variable name I considered before) on the x-axis, and then velocity on the y-axis. Again, the blue color line width is 1.5, titled 'Velocity versus Time by Differentiating Displacement Response.' Now, let me run this code. And let us check whether we are getting a similar response of velocity by different methodologies. So, Figure 3 represents multiplying the displacement transfer function by S, and Figure 4 shows the direct differentiation of velocity (dv equals dx/dt). So here, we observe that we get a similar response. So, there are different methodologies to obtain velocity; you can use either of these, as both methods are correct. Now that we are done with velocity, let us go ahead with another approach to finding position and velocity. Another methodology is the state-space approach. So, what I will do is rewrite Equation 11. Here, let me go to the next page:

$$\ddot{y} + \frac{c}{m}\dot{y} + \frac{k_s}{m}y = \frac{1}{m}f$$

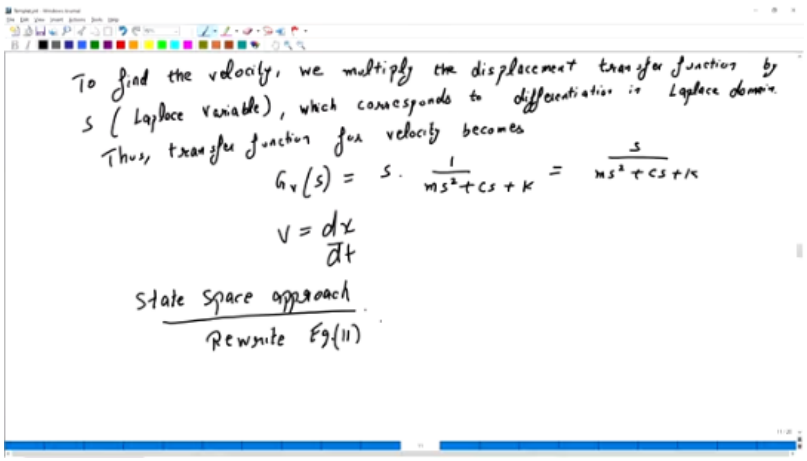
So, the system is second order. This further means that the system involves two integrators. We know that a second-order system can be represented by two first-order differential equations. So, let us define the state variables for those. x_1 and x_2: let us define

$$x_1 = y$$

$$x_2 = \dot{y}$$

What we'll do is differentiate these two equations.

(Refer Slide Time: 18:49)

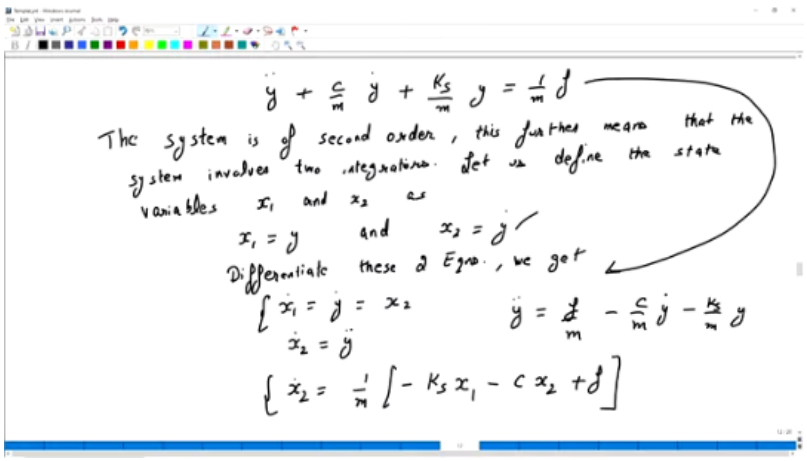


$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = \frac{1}{m}[-k_s x_1 - c x_2 + f]$$

Now here this $\dot{x}_1$ and $\dot{x}_2$ that is there we are two integrators. We can also represent this in a block diagram. That is let me just construct a block diagram for our transformed variables. So we have to integrate it twice.

(Refer Slide Time: 22:28)



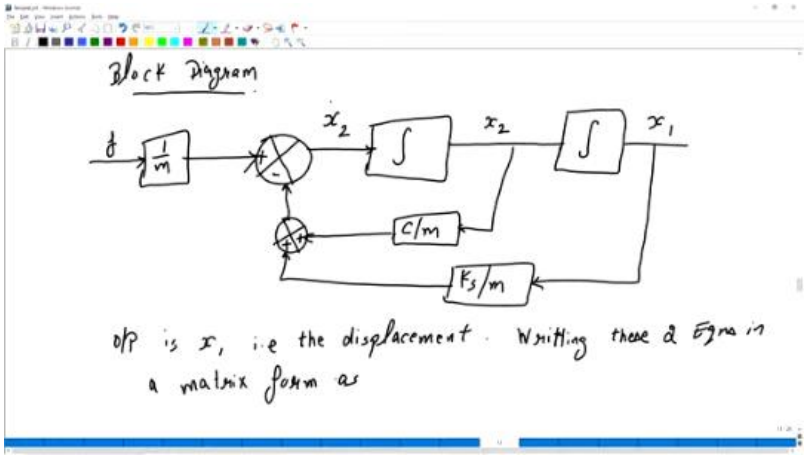
So this is the integrator symbol. We have to integrate it twice. Here it is $\dot{x}_2$, and after it is integrated, it is actually x_2, and after one more integrator, we get x1. If you see here, integration of x_2 yields x_1. Integration of x_2 yields y, and y is nothing but x_1. Here we have a summing block, so what is $\dot{x}_2$ here?

$$\dot{x}_2 = \frac{1}{m}[-k_s x_1 - c x_2 + f]$$

So here we have a block here, 1 by m. Let me write here: this is plus. Let me write: this is minus. Here we'll take this signal c by m into x_2. Here we have c by m into x_2. I'm just writing here c by m, and then x_2 is multiplied here. This is another summing block. I will consider this as plus, and I will consider this as plus. And another variable x_1 that we have here: ks upon m into x_1. So I will just multiply here with a block k_s upon m, which is x_1. x_1, and finally this goes to this summing block, and here it is minus because here we have minus c by m into x_2 and minus ks by m into x_1. This is the input here, where input is multiplied by 1 by m, and this is the force. Now this transformed system $\dot{x}_1$ and $\dot{x}_2$, where there are two integrators.

So, it is represented by a block diagram. Now, going further here, one thing I forgot to mention is that the output is x_1, which is the displacement. So, we can write these two equations in matrix form. Matrix form as $\dot{x}_1$ and $\dot{x}_2$. Here we have 0, 1, minus ks by m, minus c by m into x_1, x_2, plus 0, 1 by m to f

(Refer Slide Time: 26:09)



So, if you see this matrix form carefully, these are the same equations:

$$\begin{bmatrix}\dot{x}_1 \\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1 \\ -\frac{k_s}{m} & -\frac{c}{m}\end{bmatrix}\begin{bmatrix}x_1 \\ x_2\end{bmatrix} + \begin{bmatrix}0 \\ \frac{1}{m}\end{bmatrix} f$$

$$Y = CX = [1 \quad 0]\begin{bmatrix}x_1 \\ x_2\end{bmatrix}, \qquad D = 0$$

So, we have got A, B, C, D matrices. We know that the system state-space form is represented as

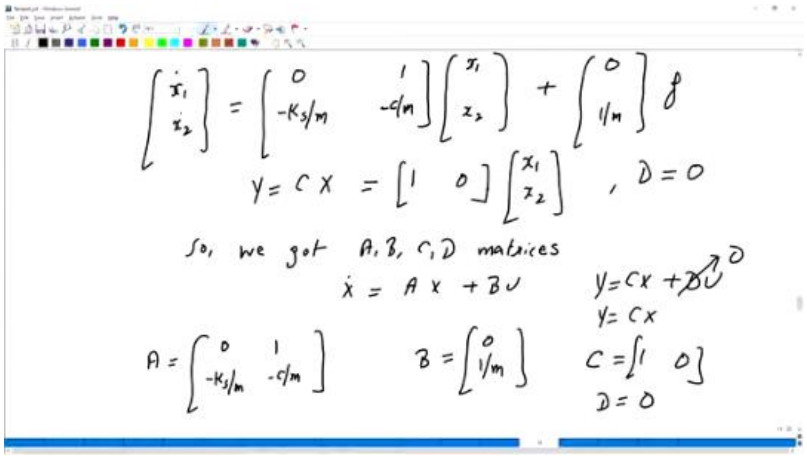
$$\dot{X} = AX + BU$$

So here, we have $Y = CX$, and this is 0. Now, here, matrix A is

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{k_s}{m} & -\frac{c}{m} \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} \qquad C = [1 \quad 0]$$

and of course, D is 0. Now, let us obtain the plots for Now, let us obtain the plots for position and velocity using the state-space method in MATLAB. So, we have plotted till figure number 4. The next is the state-space approach.

(Refer Slide Time: 28:43)



So, here a is 0, 1, and the next row is minus k by m, minus c by m. 0, 1, minus K by M,. D is 0, 1 by m, 0, 1 by m. C is 1, 0. C is 1, 0. And D is 0. And D is 0. Now here, the state space system can be formed by a syntax known as SS. Again, for more information about this syntax, type 'help SS'. So use SS to create real-valued or complex-valued state space models or to convert dynamic system models to their state space model form. So the syntax is SS(A, B, C, D). Let me show you what it actually runs. So if I run this code in state space,

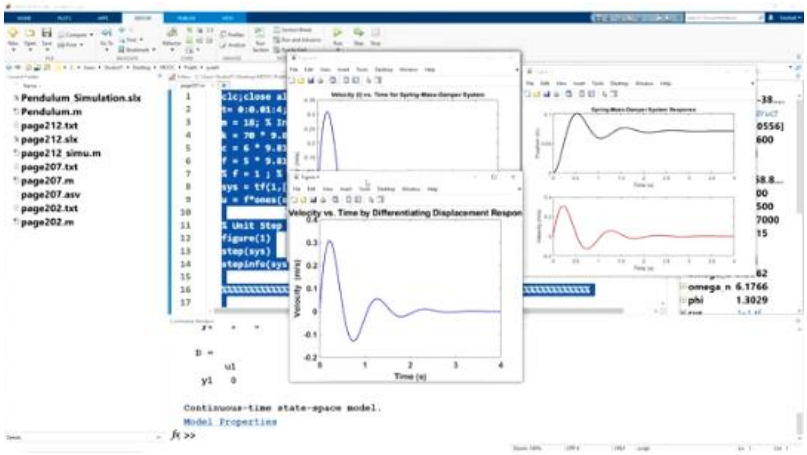
Here, MATLAB identifies the A matrix, B, C, and D. This is a continuous-time state space model. After this, we use the lsim command as was discussed earlier. Here, the output we have is Y. I've defined the variable as Y_SS. As a time variable. And one more variable I have included here.

X. I will let you know in a minute. And the syntax remains the same. The system underscore SS. Under comma U comma T. So this is the syntax for using. Lsim command. Then next. I am plotting Figure 5. So in Figure 5, I am plotting both. The position and velocity. In order to avoid a cluster of figures. So I have used the subplot command here. So here, subplot 2 comma 1 comma 1. So here, 2 represents the number of rows, 1 is the column, and the 1 represents the first figure. So here, plot t comma t underscore ss comma y underscore ss, that is x comma y. Here I have represented as minus k the lines, and the line width is again 1.5. I have commented this out for now.

I will explain in a minute. The X-axis represents time. The Y-axis represents position. The title is 'Spring Mass Damper System.' The grid is on. The second plot in the same figure is subplot (2, 1, 2). Here, 2 refers to the second figure. So here, plot T_SS. This is X(2). Now, X(2) actually represents velocity. If we had taken X(1), it would again be displacement. That is why I have considered here one more vector, x. So, here x is a vector containing position, displacement, and velocity. Now, if you run this code for Figure 5, let me run the whole code again. Right. So, this is Figure number 5.

Here we have position. This was velocity. Let me just rearrange it. So here we note that we state space method also. There is another methodology, the state space method, which is giving similar results for position and velocity. So there are different methodologies. You can opt for any, whichever you are comfortable with. Now, we again observe here that both are similar plots, displacement as well as velocity.

(Refer Slide Time: 32:33)



Now, let us proceed further. Now, let us try to change the step input at 2 seconds. Let me write it here. So let us try step input at two seconds. At two seconds. Let us consider this

as the first case. The step input here will be something like this. So this is actually two seconds. And the force is the same, that is 5 kgs, 5 kg force. And the second case will be impulse input, impulse input till 0.1 second with a force of 50 kgs this time. So, here till 0.1 second it will be 50 kgs and then it is 0. So, this is 0.1 second and this is the force is 50 kgs. Now, how to plot these responses here in MATLAB? Let us see that. So here I have defined step force input. Time, I am defining it again. Linspace, there is another syntax for defining a time variable. So this is time from 0 to 10 seconds with 1000 points. Then F force, I have written as the variable as F underscore step.

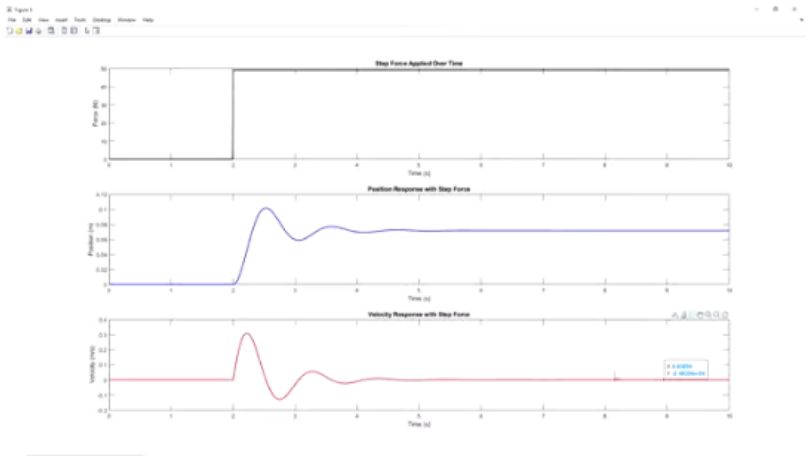
That is 5 kg force multiplied by 9.81 in newtons. and the input this input needs to be applied that is force into when the time is greater than equals to two seconds so force into time greater than equals to two so step four starting at t equals to two seconds now again the step remains procedure remains same simulate the response using l sim i have represented the variable with step response for this condition case s system underscore step ss of a comma b comma c comma d create state space system for step force and again this is the output y step t step and x step and these are the inputs l sim u step and time and this i have considered as figure six here i have considered three three rows and one column and this is the first figure when the first figure i am representing t comma u so in x axis i have time in y axis i have input u to check whether i'm getting the intended input or not k comma line width 1 comma 5 rest of the things remain same in subplot 3 comma 1 comma 2 this is the second figure of three figures sub figures and here it is plotted between time and the y step that is fun one variable First vector of this y step and the third is the velocity x underscore step into 2.

Now this of course will be second element of this x step. You try with by entering 1. So what you will actually observe is that is again the displacement vector. So if you want a velocity vector you have to mention the second element of that vector. Now let me plot this. I am re-running the entire code. Now see here, this is the step input, step input force which is applied at 2 seconds and this is at 5 kgs, that is 50 N force. So this is the input that we actually intended for. And see this position and this is the position and this is the velocity response for mass spring damper system. So the position it is settling around 0.07 approximately 0.07 meters and here the velocity is again going to 0.

So this was figure 6. Now let us proceed for an impulse response. How we can write a code for that. impulse force input so here impulse force magnitude is i've assumed it as 550 into approximately 10 500 newtons and duration is 0.1 second so 0.1 second because the force is actually very high force for impulse in newtons and this is only for short

duration so i have defined next as defined impulse force as a short pulse so this input force will be applicable when impulse 500 when the time is less than or equals to the intended time duration, which is 0.1 second.

(Refer Slide Time: 38:08)



Then again, the process remains the same. Simulate the response using the state-space SS syntax. Create a state-space system for the impulse response. And these are the outputs. Y impulse, T impulse, and X impulse.

Lsim. And the rest of the things remains the same. Now, let me plot this. Let me run the code again. Now, this is Figure 7. Now, see here the force is actually 500 newtons for the first 2.1 seconds, and then it is 0. So here again, we observe that since the input is actually 0 here, the displacement comes down to 0, and the velocity also comes down to 0. So what we can actually infer from this is that with these plots, we can easily say that the system is stable, and there is another methodology also that we can infer the system stability. If you remember, we have obtained a matrix, and we know that the system stability could have been easily checked and assessed by checking eigenvalues. Eigenvalues of the A matrix since we have already obtained the eigenvalues of the A matrix. Now, before we check for the roots, we know that for the underdamped case. So, for the underdamped case, we know its zeta should be less than between 0 and 1. Let us check how much zeta is. Here, zeta is 0.2647, which is between 0 and 1. So, the roots will obviously be complex conjugate. So, we can check this directly again in MATLAB.

I have already entered the command. The syntax to check the eigenvalues is EIG of A, and here we obtain -1.635 plus or minus 5.95. 6i. Here, since the real part is negative, without plotting anything, we can easily say that the system is stable. This further means

that any bounded initial disturbance will eventually decay to zero over time. That is what we checked in these figures when there was an initial disturbance or a force for 0.1 second, which is an impulse force. The system is coming back to its equilibrium position. Similarly, in Figure 6, with a different control input, a different step input, the system is again coming to its equilibrium point, and the velocity is coming to 0. So, learners are encouraged to plot these responses by themselves. So, with this, we conclude the topic on simulation of a mass-spring-damper system in MATLAB. In the next lecture, we will take a different example of a pendulum system, but this time we will simulate it in Simulink. Thank you.

(Refer Slide Time: 43:57)