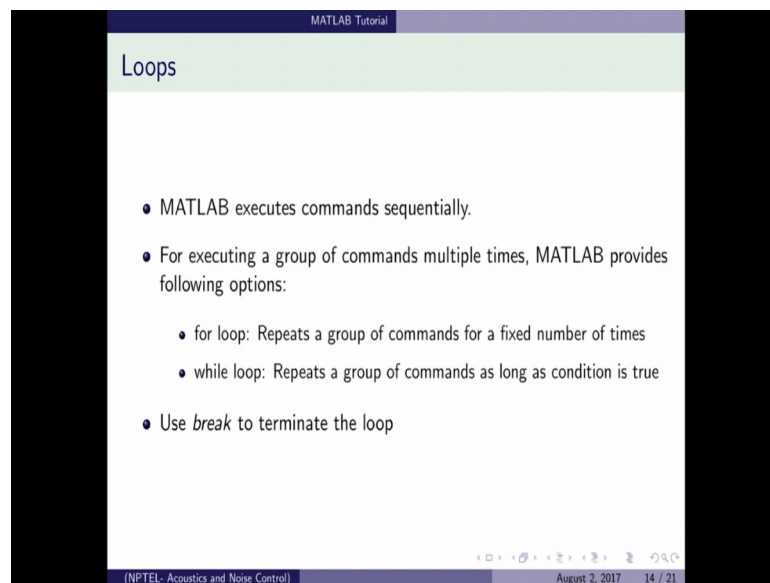


Acoustics & Noise Control
Dr. Abhijit Sarkar
Prof. Ajinkya A Baxy
Department of Mechanical Engineering
Indian Institute of Technology, Madras

Lecture – 42
MATLAB Tutorial - 4

(Refer Slide Time: 00:17)



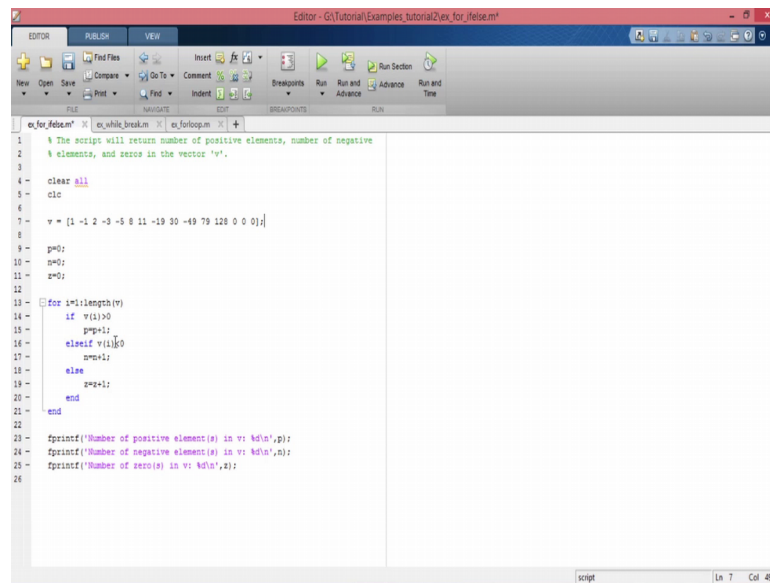
The image shows a slide from a MATLAB tutorial. The slide title is "Loops". The content includes the following bullet points:

- MATLAB executes commands sequentially.
- For executing a group of commands multiple times, MATLAB provides following options:
 - for loop: Repeats a group of commands for a fixed number of times
 - while loop: Repeats a group of commands as long as condition is true
- Use *break* to terminate the loop

At the bottom of the slide, there is a footer that reads "(NPTEL- Acoustics and Noise Control)" on the left and "August 2, 2017 14 / 21" on the right.

Let us now move on to Loops. Loops help us to repeat the same process or to execute the same statements repeatedly. MATLAB provides 2 options for looping; one is for loop and other one is while loop. The difference between the 2 is their nature; for loop repeats the commands for a fixed number of times whereas, while loop will keep on repeating until the condition becomes false. We will later see that we can sometimes avoid Loops and increase the computational speed by vectorising our code that we will see later.

(Refer Slide Time: 01:07)



```
Editor - G:\Tutorial\Examples_tutorial2\ex_for_forloop.m
EDITOR  PUBLISH  VIEW
New Open Save Compare Go To Comment Breakpoints Run Run and Advance Run Section Run and Time
FILE NAVIGATE EDIT BREAKPOINTS RUN
ex_for_forloop.m ex_while_break.m ex_forloop.m
1 % The script will return number of positive elements, number of negative
2 % elements, and zeros in the vector 'v'.
3
4 - clear all
5 - clc
6
7 - v = [-1 -1 2 -3 -5 8 11 -19 30 -49 79 128 0 0 0];
8
9 - p=0;
10 - n=0;
11 - z=0;
12
13 - for i=1:length(v)
14 -     if v(i)>0
15 -         p=p+1;
16 -     elseif v(i)<0
17 -         n=n+1;
18 -     else
19 -         z=z+1;
20 -     end
21 - end
22
23 - fprintf('Number of positive elements in v: %d\n',p);
24 - fprintf('Number of negative elements in v: %d\n',n);
25 - fprintf('Number of zeros in v: %d\n',z);
26
```

Let us take an example for for loop. So, let us visit our good old distance code, let us say I want to find the distance between the 2 points again, but this time the x naught coordinates is wearing 10 times instead of calling the function 10 times, let us see how we can automate the process and we can use for loop. For that first, we will describe the code. Let us clear the work space command window and let us say x naught is 1 to 10, y naught is 2 x 1 is 3.5. Let us say and y 1 is 5.

So, now we have to run our for loop 10 times, since there are 10 elements in this vector. So, we have to find the distance between the 2 points x naught moves 10 times. So, we will write start for loop with for, then we will set our iteration variable or loop counter as we call it, I will start from 1 up to length of x naught. Here again, we have used the colon operator, as you have observed also need not necessary, the increments should be one, you can even put 2, but in this case due to requirement, we will write one column length x naught. Now for each iteration; we will store the distance in vector r.

So, r of I will store that number in ith position in each iteration. So, for first iteration, the solution will be stored in first element of r, in the second iteration, it will be stored in the second element and so on. So, r i will be square root of x naught minus x naught of i, we have to extract i th element of x naught minus x 1 square plus y naught minus y 1 square and we will tell MATLAB for loop has ended with keyword end so let us save the program, now let us run it.

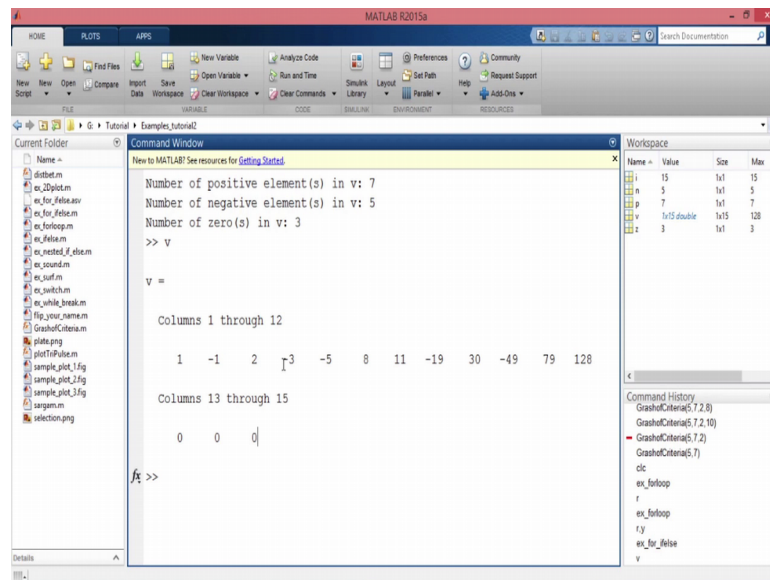
Since we have saved our program in a separate folder, but the current work directory is not that folder. So, MATLAB is asking us to change the current directory to the folder where the program is saved. So, if we say change folder MATLAB will automatically do it for us and as you can see MATLAB is calculating r and these, this is how the distance between the 2 points will vary as x naught will change from 1 to 10, there is 1 more thing which we can do we can calculate r in 1 more way by using the function `dist` between points. So, let us say its r and here we will write.

So, we will say x naught is x naught of i y naught x 1 and y 1. Let us complete both and will compare it later. So, let us run the program and let us see r and y . So, as you can see, either we can use this function to calculate the distance or we can directly use the distance formula we can use if else statements in Loops 2. So, let us see an example where we can use if else statements in the for loop the script will return number of positive elements number of negative elements and 0s in the vector v , here we defined our vector v and initialized scalars p n and z .

So, p will store number of positive elements in v n will store the negative elements and z will store the number of 0s. So, we will run our loop for length of v number of times. So, that all the elements in v are covered, then we will use if statement to see whether the i th element in the v is greater than 0 or is it less than 0 and the third possibility remains is it equal to 0. So, what happens if let us say in first iteration when i is 1. So, v of one is one. So, MATLAB will check is v of one greater than 0 yes. So, p will be updated to one p will be p plus one and then it will exit the if else statement go to end again go to for loop then in the second iteration v of 2 will be minus one

So, MATLAB will check is v of 2 is greater than 0 no. So, it will check is v of 2 less than 0, yes. So, it will add 1 to n and again, it will exit if and it will go to n and again the for loop continues until we reach the end of the vector v then finally, we will print our result using `printf` command. So, again it will tell us number of positive elements in v percent d because it will be an integer and we have to mention p n and z . So, this is how the code will work let us see. So, we will save it and let us run the code.

(Refer Slide Time: 09:27)



So, MATLAB has already printed the result that number of positive elements in v are 7 and negative elements are 5 and there are 3 0s. So, let us count them let us count 0s, first there are 3 0s, yes, let us count number of negative elements minus 1, 3, 5, 1, 2, 3, 4 and 5. So, you see how it works, similarly you can combine if else statements which statements with for Loops and create some interesting Loops similar to nested if else statements we can have nested for Loops also let us take an example of nested for loop.

So, given a vector of length p, we will construct a matrix of size n cross m, search that the product of n and m is p that is the number of elements in the matrices should be equal to number of elements in the vector which has length p. So, here we have already written the program. So, let us go through it as usual, the first line is the description of the code, then this p will define the length of the vector v. So, that is how many elements vector v will contain, now say for example, we have taken p to be 6. So, the matrix size can be either 2 cross 3 or 3 cross 2.

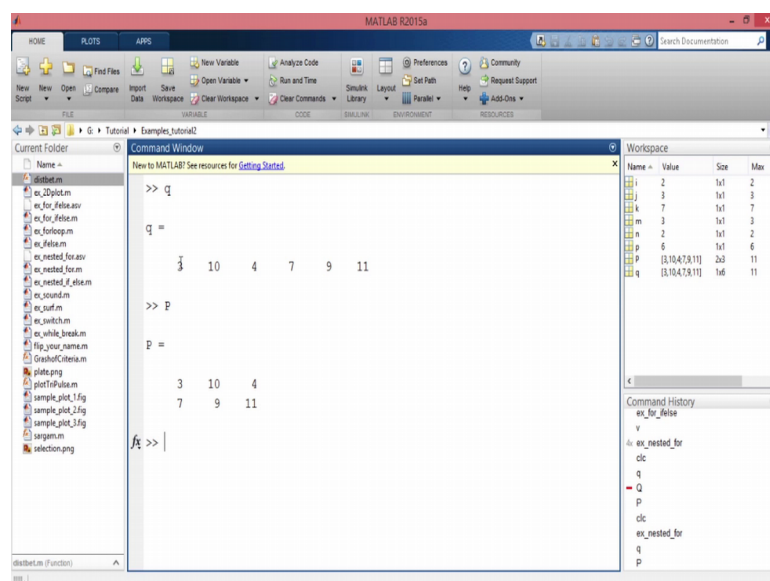
Now, we will construct a random vector of integers. So, for that command in MATLAB is r and I and then we have to give 3 inputs. So, the first input is the range. So, here I have written times p. So, the command rand I will create a vector of random integers between one and 2 p. The next 2 numbers 1 and p will indicate the size of the vector of the matrix. So, here since we want row vector, I have written 1 row and p columns. So, q will contain a matrix of random integers from 1 2 to p and it will be of size 1 cross p.

We will then initialize a variable k as 1, we will see how that is useful later, now, we will write the outer for loop, outer for loop will begin from 1 to n and then the inner for loop will begin from 1 and it will run a for m number of times and then we will say that p of i comma j is q k that is in each iteration p of element in the ith row and jth column will be kth element in the vector q, let us try to run this ourselves. So, let us say for i equal to 1, we enter this for loop outer for loop, then again we enter this inner for loop when j will start iteration.

So, j will be equal to 1. So, so in this line we have p of 1 comma 1 equal to q of k, if you do not initialize k MATLAB will give us an error. So, we have initialized k as 1. So, in the first iteration we have p 1 comma 1 is q of 1 after that k will add 1 to itself and become 2. So, in the next iteration j will become 2 and we will have p of 1 comma 2 as q of 2, again k will update itself to 3 and j will become 3. So, p of 1 comma 3 will be q of 3 now, what happens k will update itself to 4, but since m is only 3 j this inner for loop will run only 3 times.

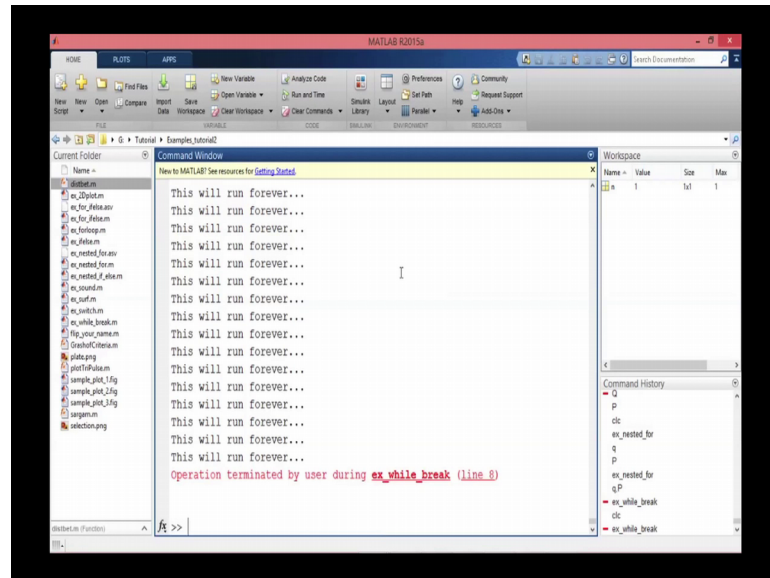
Since the counter has counted 3 times, we will exit this inner for loop and we will go to outer for loop. Now remember that k has value 4 and outer for loop will run for 2 times because the counter is set for 2. So, outer for loop will run once again, now the value of I will be 2 again we will run this loop 3 times and k will update itself in each iteration, we will save this code and let us run it.

(Refer Slide Time: 14:55)



So this is our vector q , it contains 6 elements and this is our matrix p . So, as you can see the elements in q has been arranged in a rectangular matrix fashion the size of q is 1 cross 6 and the size of p is 2 cross 3.

(Refer Slide Time: 15:24)

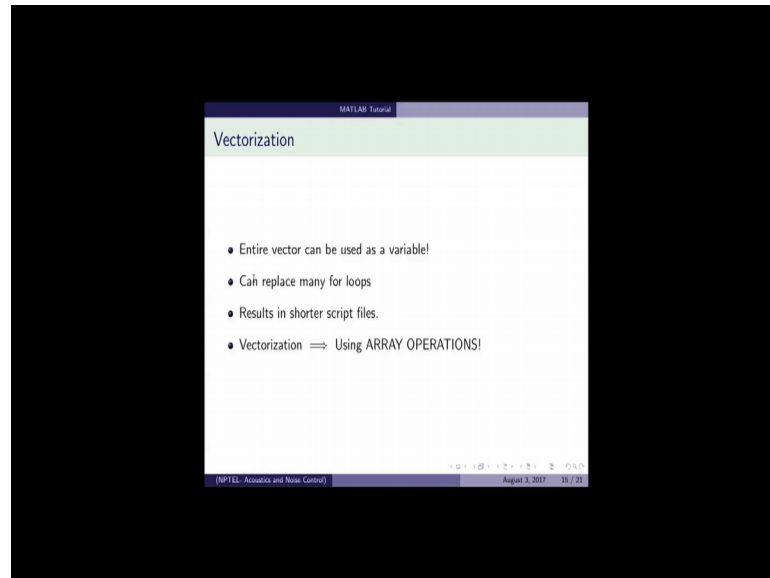


Similarly, if we want as matrix of size 3 cross 2, we just have to change m and n and now you can check p is of size 3 cross 2. So, this is u and p . Now we come to while loop, as we have seen, while loop repeats the action as long as the condition is true in other words. We can use while loop when we do not know how many times, we have to repeat the action that is when the count is unknown. So, let us see an example of while loop. So, if you are as careless as i and if you define n to be 1 and you said condition on while loop that perform this action as long as n is less than 2 which is always true.

So, MATLAB will get stuck here and it will play print this string forever. So, as you can see MATLAB is terminated the operation when you press control C. So, in this way, we can get out of an infinite loop, the smarter way would be not to get stuck in a infinite loop. So, when using while loop, care must be taken that we should not use such conditions due to which we may get stuck into an infinite loop. We can also use break keyword to terminate such infinite Loops after a particular range, suppose, I want to run this, let us say we will given print number of times it is running. So, percent d and that will be m let us initialize m , m is 1 and we will increment m by 1 in a iteration then they will set a condition that then m will cross 50 break from the loop come out of the loop.

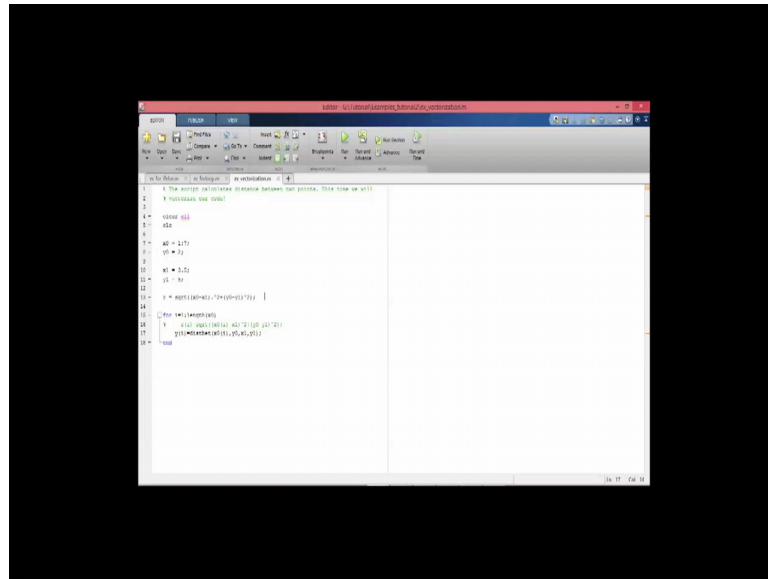
So, let us save and let us go to command window. So, as you can see here MATLAB has printed this 50 times then m will increment itself to 51 then MATLAB check this condition is m greater than 50, yes. So, if yes it will break the loop and it will come out.

(Refer Slide Time: 19:02)



Now, let us see what vectorization means if you see all over previous examples you will notice that we are working with one number at a time which essentially means that we are working with scalars take a look at this example we access 1 element from the x naught vector in each iteration that is we use scalar as a variable and to automate the process, we used for loop by vectorising we can use the entire vector as a variable vectorising has its benefits, we replace many for Loops and if else statements if we can victories our code. So, this results in shorter quotes which increase credibility of our codes and debugging becomes quite easy.

(Refer Slide Time: 20:22).



```
1 % The script calculates distance between two points. This time we will
2 % calculate the dist
3
4 % Using for loop
5
6 x = 1:7;
7 y = 2:5;
8
9 % For loop
10 for i = 1:length(x)
11     r(i) = sqrt((x(i)-x1)^2 + (y(i)-y1)^2);
12 end
13
14 % Using vectorization
15 r = sqrt((x-x1).^2 + (y-y1).^2);
```

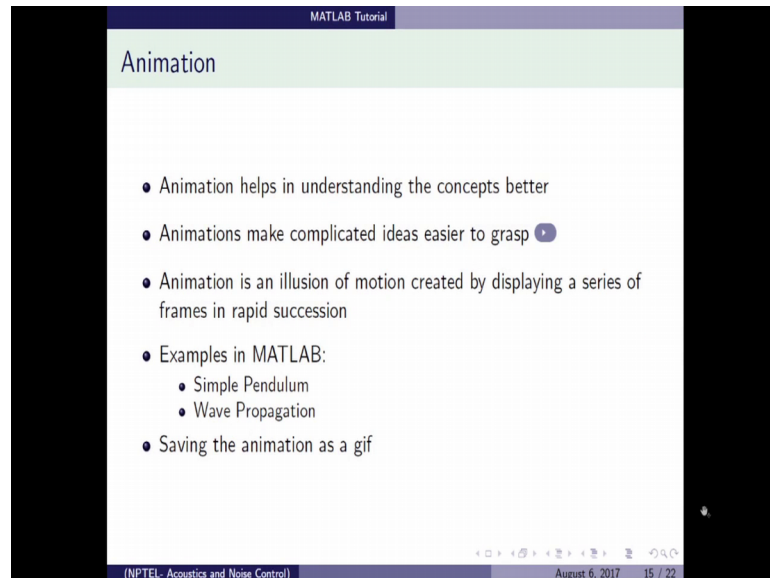
Let us solve the same example by vectorising, now you will ask what actually means vectorization well we already know it, the vectorization basically means using array operations whenever we can; let see the same example distance between the points. Now using vectorization as usual the first line is information. So, let us define x naught, it varies from 1 to 7 y naught is 2 x_1 is 3.5 and y_1 is 5, we will store our result in r variable will be square root of now we will directly write x naught, we will treat x naught as a variable although it is a vector.

x naught minus x_1 , since we are dealing with array and going to square each element in the vector x naught, we have to use array operation and we will use a dot here to tell MATLAB to perform element wise operation plus y naught minus y_1 . So, let us copy this for loop from here. So, that we can verify that our r ; what we have written is actually correct. So, while we store the distance between the same points, but it will be using the function `dist` between, so, let us save our file and let us run. Now let us go to command window and let us print r and y .

So, as you can see r computes exactly the same values as y , it means that r as correct values. So, if you observed here we have replaced the entire function of for loop by just one line and we have achieved that using array operation which means we have vectorized our code. So, whenever possible try to vectorize your code, it will not only

increase the computational speed, but the code will also look nice and it will be easy for the user to read it and understand it.

(Refer Slide Time: 23:58).



MATLAB Tutorial

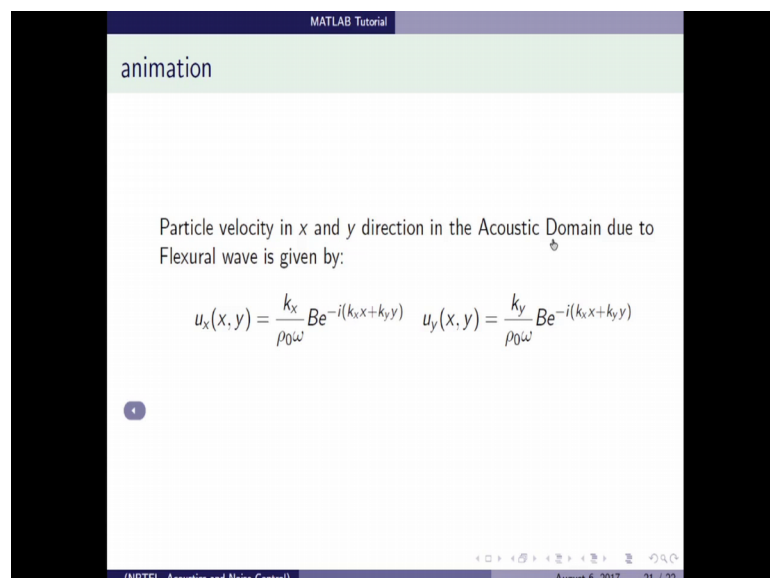
Animation

- Animation helps in understanding the concepts better
- Animations make complicated ideas easier to grasp
- Animation is an illusion of motion created by displaying a series of frames in rapid succession
- Examples in MATLAB:
 - Simple Pendulum
 - Wave Propagation
- Saving the animation as a gif

(NPTEL- Acoustics and Noise Control) August 6, 2017 15 / 22

Finally, we come to animation; here we will need most of those topics which we studied earlier. So, why do we need animation? Animation helps in understanding the concepts better, since it is a visual media, it sticks in our brain for a longer time animation also makes complicated ideas easier to grasp many times while looking at a complex mathematical expression we wonder what does these looks like physically for example.

(Refer Slide Time: 24:36)



MATLAB Tutorial

animation

Particle velocity in x and y direction in the Acoustic Domain due to Flexural wave is given by:

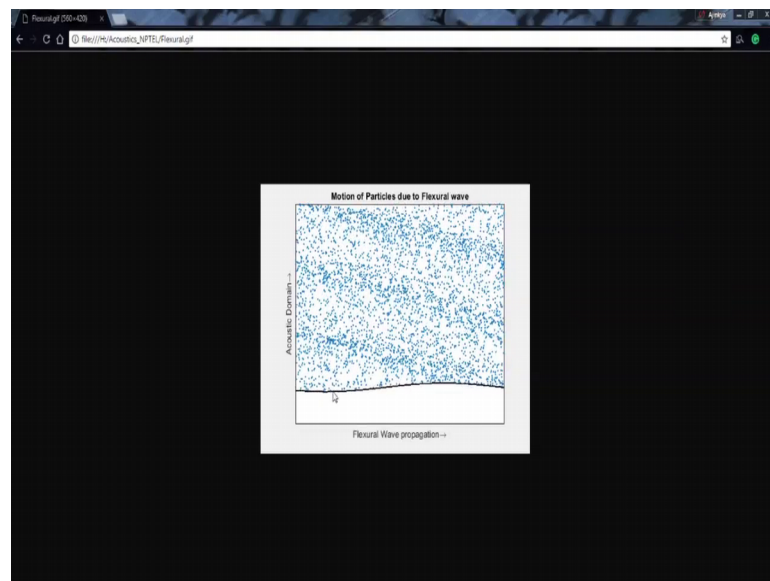
$$u_x(x, y) = \frac{k_x}{\rho_0 \omega} B e^{-i(k_x x + k_y y)} \quad u_y(x, y) = \frac{k_y}{\rho_0 \omega} B e^{-i(k_x x + k_y y)}$$

(NPTEL- Acoustics and Noise Control) August 6, 2017 21 / 22

Here is expression for motion of particles in the acoustic domain due to a flexural wave in a structure. Do not worry you will learn about this later in the course.

So, what does this equation convey? It tells us how particles will move in acoustic space, if there is a wave in the structure tough 2 image in right, let us see in animation of the same.

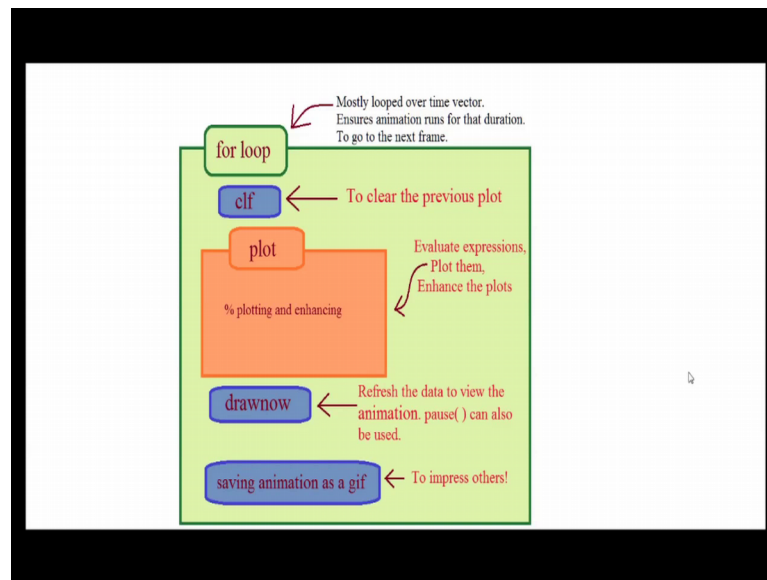
(Refer Slide Time: 25:06)



This black line indicates the flexural wave which is let us say setup in a beam and due to this forward moving flexural wave the acoustic particles in this domain have this type of motion. So, instead of just looking at this equation and trying to imagine what it looks like it would be much nicer if we can actually plot this and animate this in MATLAB.

So, how do we create an animation? Animation is an illusion created by showing a series of frames in rapid succession, suppose I want to animate a chicken crossing a road, I will take snapshots at different times then display them rapidly one after the other. So, as to create an illusion of motion we use the same technique in MATLAB, but instead of taking snapshots in the camera we use plot commands.

(Refer Slide Time: 26:09)



This can be considered as a guide to construct animations in MATLAB the figure shows essential elements required for it.

First we define the variable that we need then we need a for loop to ensure that the animation runs for the specified duration of time within calculate the time varying quantities in its iteration of for loop. For example, while animating this the time varying quantities where the displacement of particles then we plot our variables of interest, if we had to use a hold on, then do not forget the clear figure command which is in MATLAB specified by `clf` in animation hold on and `clf` go hand in hand. Now if we miss this `drawnow` or `pause` command, we will not be able to see the animation because MATLAB will finish the computations too fast.

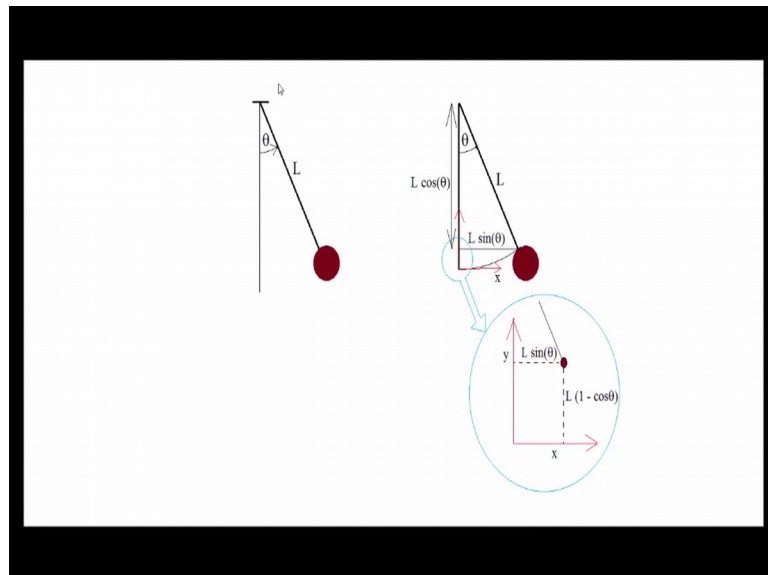
So, we need to use `drawnow` to refresh the figure data we can even use `pause` where in the parentheses we have to mention the number of seconds, MATLAB will pause before giving the control to the next line we repeat this over length of time vector finally, we can save this animation as a gif or gif files show to others and impress them.

So, let us see 2 example for animation in MATLAB first we will animate a simple pendulum. So, here is the code for a simple pendulum. So, here in this part we had defined our variables for example, the length of string acceleration due to gravity initial displacement of our simple pendulum I agree 45 degree is too much in order to visualize the animation properly, let us use 45 degrees here we convert the initial displacement

which is in degrees 2 radians then here we discretizes the time variable and it run for 2 cycles and here are the Cartesian coordinates which we will use for plotting these are the factors fine varying quantities as the displacement of the bob will change with time. Now here comes our for loop. So, first we set the counter I from one to length of t then we write $c L f$, since we are using a hold on. So, if we miss this $c L f$, but use hold on all the figures will printed each other and it will be a complete mess. So, hold on and clear figure both hand in hand within calculate the position of bob in terms of theta.

So, let us see the geometry of the simple pendulum.

(Refer Slide Time: 29:31)



So, here is our simple pendulum and theta is measured from the vertical axis the length of the string is L the mass of the bob does not matter and now we will plot our vectors in Cartesian frame. So, we need x and y component of the displacement, but as you observed here we are computing displacement in terms of theta that is in calculating what this theta will be as a function of time, but I need y and x as function of time. So, here we setup cartesian coordinate system at this point where theta is 0 and then if we magnify this here we see that this length is actually L minus L cos theta. So, this much length is L into one minus cos theta and this length is L sin theta therefore, we will calculate theta as a function of time and then we will use that theta to compute y and x. So, here we do that here we calculate theta which is of which is solution for simple

pendulum when we convert our theta into Cartesian coordinates x and y then we plot the bob will plot the motion of the bob.

These x and y are the coordinates of the bob as it moves in time and we use a marker as a bob then we plot the string since the bob is moving and the string is attached to it the string also has to move along with the bob. So, we use line command for it we could have used plot also, but line will also do. So, the syntax for line is we have to mention 2 vectors as you can see here this is one vector, this is another vector and this will contain x naught y naught x 1 y 1.

So, one vector will be for x coordinates, one vector will be for y coordinates when we plot the vertical line corresponding to theta equal to 0 and here we will print the theta values as our time progresses to display that values we use annotation text. So, syntax for text is we have to mention the x coordinates the y coordinate and the string then we said some access properties and we can either use draw now or pause. So, let us hear animation.

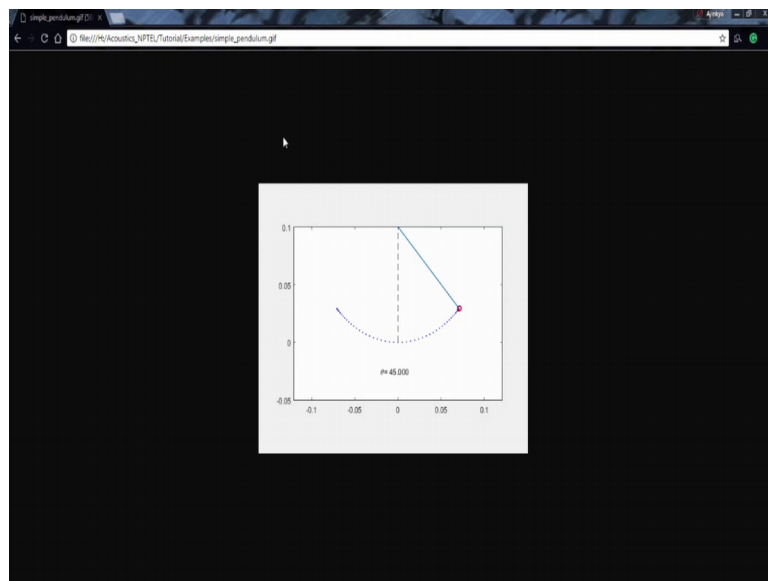
So, as you can see this x will print theta over here one thing of observed here that is while using a s print f we have use a double backslash theta. So, it should be kept in mind that if you are using s print f to print the Greek letters of the Greek symbols, we have to use a double backslash a single backslash will not work, but while using print f or in titles and all we can use a single backslash theta like that; now if we want to see the path which is traced by this bob as it moves in time. So, how do we do it, observe here that x and y are vectors. So, at each iteration x is storing the x coordinates of the bob and y is storing the y coordinates of the bob. So, we just have to plot x and y instead of x of I and y of I if we plot x and y, we can trace the path of the bob. So, let us plot it.

Color is blue and marker is a simple dot let us call this p naught; p naught is the handle of plot for the path. So, we will said p naught dot marker size to be 4. Now let us run this code and see whether we can see our path. So, why what happened why we could not see we use plot command everything is right MATLAB did not give any errors. So, still why we could not see our path the reason is replaced hold on here instead of here. So, what happened MATLAB plotted p naught then since there was no hold on and we again ask MATLAB to plot p it replaced p naught and it happened in each iteration.

So, we now rectify our mistake by putting hold on after p naught and now let us see if we get our path as you can see here is our path. Now let us see; how we can save our animation as a gif file. So, first we will define our file name and then we write the following code. So, this saving as a gif file involves 2-3 processes, first this get frame will capture the current access as a movie frame. So, what happens when we plot all this we will generate a plot. So, this gate frame will grab that plot and save it as a frame as a movie frame next step we convert that frame to an image using frame to i m function, this frame to i m will return the image data associated with the movie frame after that we will convert the r g b image to an indexed image and finally, we will write that image to graphic file using I m write command.

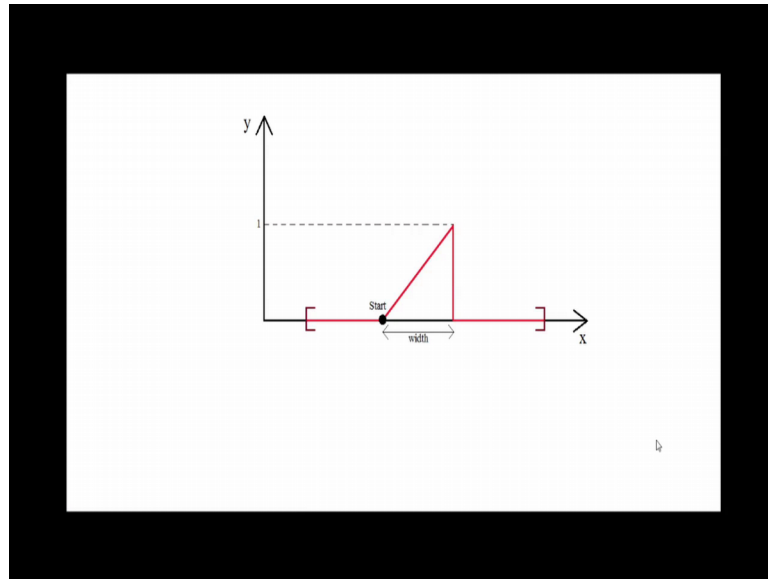
So, these are the steps involved in converting the frame into image and finally, we will use for loop and this if else statement and I m right to save the file as a gif. Now let us save and run our code and see if our gift is created.

(Refer Slide Time: 37:57)



The file will be saved in the current work directory let us see. So, as you can see we have successfully saved a gif the next example which we will see is of wave propagation. So, there is a triangular pulse which will move forward with time. So, let us first see the problem.

(Refer Slide Time: 38:53)

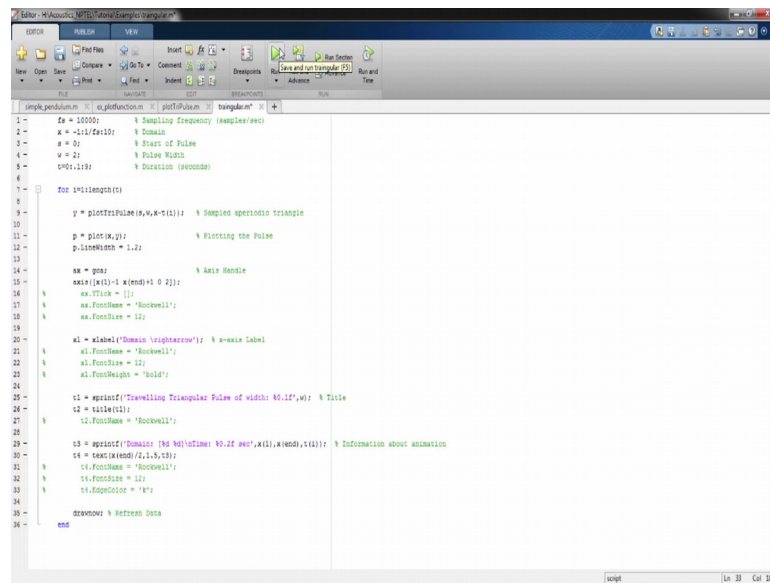


So, this is how our pulse will look like a triangular pulse. So, first what we will do is we will write a function to plot this pulse.

So, we will ask 3 input from the user will ask user the domain of the pulse as you can see it is enclosed in the square brackets, then we will ask user where the pulse starts. So, we will ask user start, we will ask what is the width of the triangle or width of the pulse we will ask user the domain start and the width the pulse will always be normalized. So, that whatever start or width or range the user gives the height of the pulse will always be unity. So, let us see a function for that. So, this is the function.

So, this is the function in which will plot our pulse. So, this is the definition line 3 inputs are expected to start the width and the length and we have one output parameter y this is the help available for the function. So, let us try one thing let us search our own function which we have written here Plottri.

(Refer Slide Time: 47:31)



```
1- fs = 1000; % Sampling Frequency (samples/sec)
2- x = -1:fs/100; % Domain
3- v = 0; % Start of Pulse
4- w = 2; % Pulse Width
5- t=0:1:10; % Duration (seconds)
6
7- for i=1:length(t)
8
9- y = plotFnsPulse(v,w,w-t(i)); % Sampled aperiodic triangle
10
11- p = plot(x,y); % Plotting the Pulse
12- p.LineWidth = 1.2;
13
14- ax = axes; % Axis Handle
15- axis([x(1)-1 x(end)+1 0 2]);
16- %
17- % ax.FontName = 'Helvetica';
18- % ax.FontSize = 12;
19
20- %
21- % xl = xlabel('Domain (rightarrow)'); % x-axis Label
22- % xl.FontName = 'Helvetica';
23- % xl.FontSize = 12;
24- % xl.FontWeight = 'bold';
25
26- %
27- % t1 = sprintf('Triangular Triangular Pulse of width: %0.1f',w); % Title
28- % t2 = title(t1);
29- %
30- % t3 = sprintf('Domain: [%d %d]Time: %0.1f sec',x(1),x(end),t(i)); % Information about animation
31- % t4 = text(x(end)/2,1.5,t3);
32- % t5.FontName = 'Helvetica';
33- % t5.FontSize = 12;
34- % t5.EdgeColor = 'k';
35
36- drawnow % Refresh Data
37-
38- end
```

So, as you can see MATLAB will print exactly those lines which we mention here. So, for other users to understand your function how to use it and all always mention the purpose and how to use it.

So, was the user has passed 3 values, we will store the range in variable x, we will store the start in variable v and we will store the coordinate of width in w. So, what I mean exactly is this even though the width is this we need this coordinate. So, what we will do is we know the start coordinate we will add width to it to get the width coordinates. So, that is why here, we added start coordinates to the width now we will use for loop in order to generate data in this range. So, we will run for loop for the length of this discretized x, we will discretize our domain into small intervals and then we will run the for loop over the length of this vector.

So, essentially the user must give range as a vector then we come to the if else statement. So, why do we need if else is that there is condition for plotting before this start coordinate the y values should be 0 and after this width coordinates the y values should be 0 and in between this start coordinate and width coordinate y of x will be a straight line. So, we achieve that using and if else statement; so, in this if statement check condition whether our x of i in the ith iterations is it greater than v and less than w that is it lying within this range if it is then, we calculate y using this formula if it lies outside this range then we set y of I to be 0 finally, we plot x and y. So, let us try.

So, here how we will do it n is the sampling frequency that is we have sample x into these many samples the range we have given is minus one to ten. So, the pulse domain will be minus one to 10 the pulse will start at coordinate one and the pulse width will be 3. So, what we expect if we run this code is that the domain of the pulse will be minus one to 10 the pulse will start from one and the pulse will end at 4.

Since the width of the pulse is 3. So, let us see as you can see here the domain of the pulse is minus 1 to 10, it starts at 1 and ends at 4. Now let us play around; let us set starting to 2 and let set width to 3.5. Now let us run the code again and again in the start is at 2 and the end is at 5.5. So, that the width of the pulse is 3.5 and observed that the height is always one. So, the height is always one this time the width is 2. So, this is our function which plots the triangular pulse.

Now, let us come to the animation part. So, here again x is the domain observed that we used `linspace` command here to discretize x and here we have used colon operator and the start of the pulse is at 0 the width of the pulse is 1.5, let us make it 2 and we want our simulation to run for 9 seconds with increment of 0.1 seconds. Now we plot our forward moving where here since we are plotting here, let us comment the plot command in the function, let us save the function come let us come back to the code. So, here we store the output of this function we have a output parameter here.

So, we store the output of this function into y which is a basically information about the pulse and x minus t , I will ensure that the pulse is moving forward in time then we plot the function y verses x here y is the time varying quantity then we get the current axis handle in x .

For animation purpose do not forget to set the axis because if you do not set axis in each iteration MATLAB will adjust on its own and the animation will look like a complete mess. So, one previous thing which we learnt was to use `hold on` and `clear figure` hand in hand the second thing to keep in mind is always provide axis limits.

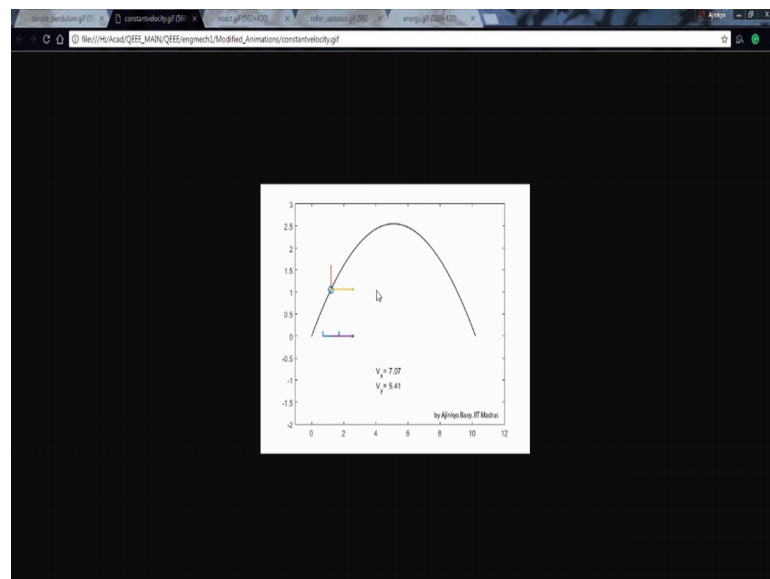
First let us comment this and see how animation is and. So, that we can appreciated later the decoration is off. So, we have set the axis limit we have labeled the x axis we have labeled the title and we have placed a text which tells the domain and the time. So, let us run the code as you can see the code has run for nine seconds it has told me; what is the domain of my pulse and it is also told me; what is the width of my pulse. So, as you can

see we have some peaks here we are only interested in the x ticks. So, that we can know what the domain of my pulses we do not want to see this, right, if we are sure that each time in the pulse is normalized. So, amplitude will be one. So, how do we set this off?

For that we use this command `y tick off` control `t` is the shortcut to uncomment control `r` to comment, then we set the font name and font sizes for uniformity. Now let us run the code again observe the y axis, this time as you can see the y axis labels are not visible and the x axis labels have change the font and the figure in all looks quite uniform. So, we have seen 2 examples of animation we have seen how to save animation as a gif, we know what are the important factors that should be considered while doing an animation in MATLAB keep that guideline in mind when you do the coding do not forget simple things like axis limits clear figure hold on. So, those are simple things and we tend to miss them and then it will screw up entire animation.

So, we have seen 2 examples of animation, the structure will be the same only things changing will be the expression which we will use to calculate the time varying quantities rest everything the structure will be same to save the file or for loop structure etcetera for practice you can take few concept from vibration or engineering mechanics and try to code them and create animations for example.

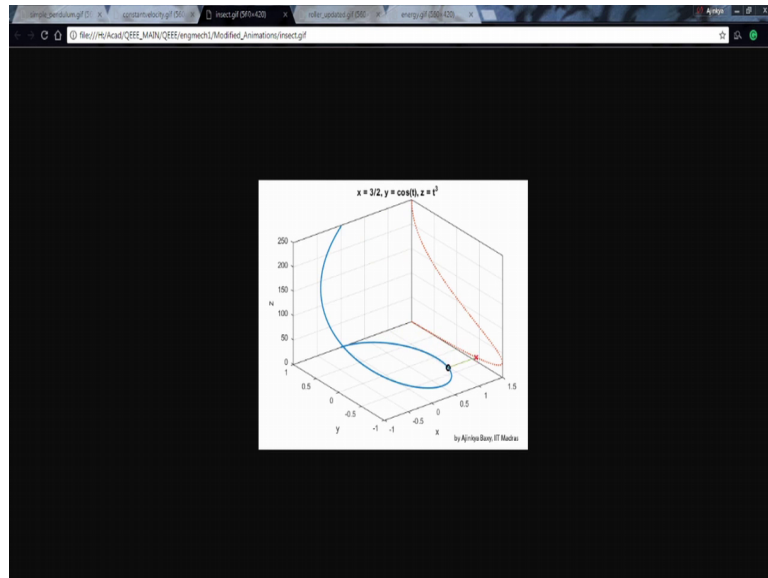
(Refer Slide Time: 50:31)



This animation will show that the x component of the velocity during the projectile motion will remain same where as the y component will change over time. So, here we

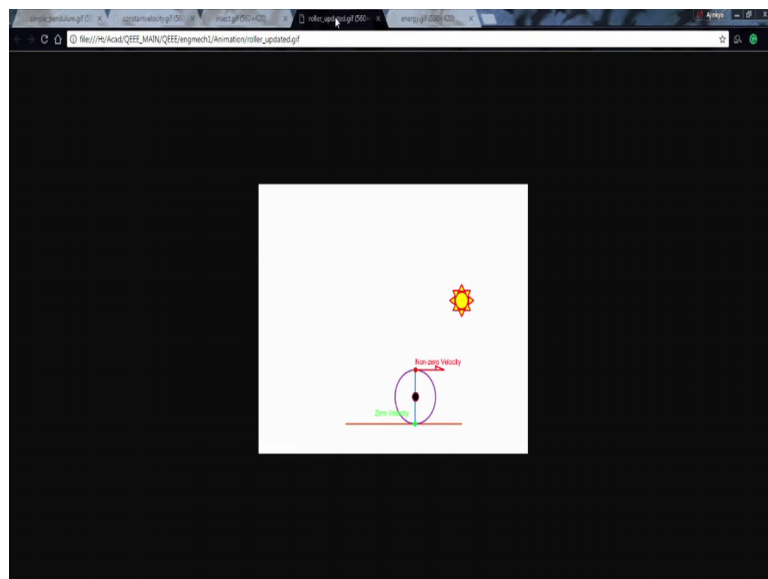
can see the y component is going down, down and down. So, that finally, it will go to 0 and then the gravity will pull it downwards and the velocity again starts increasing. So, for these arrows I have used quiver plot you can check them in MATLAB help.

(Refer Slide Time: 51:11)

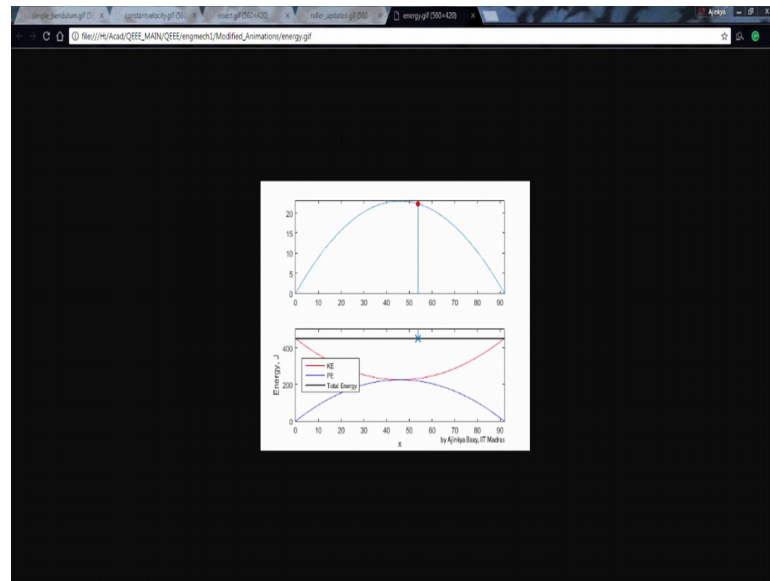


Next, this is a 3 d plot which has been a plotted using command plot 3. So, it shows some curve and a particle is moving along it whose projection can be seen as this red dot this is example of rolling motion the point at which it touches ground has 0 velocity that is what we can see finally about this energy.

(Refer Slide Time: 51:34)



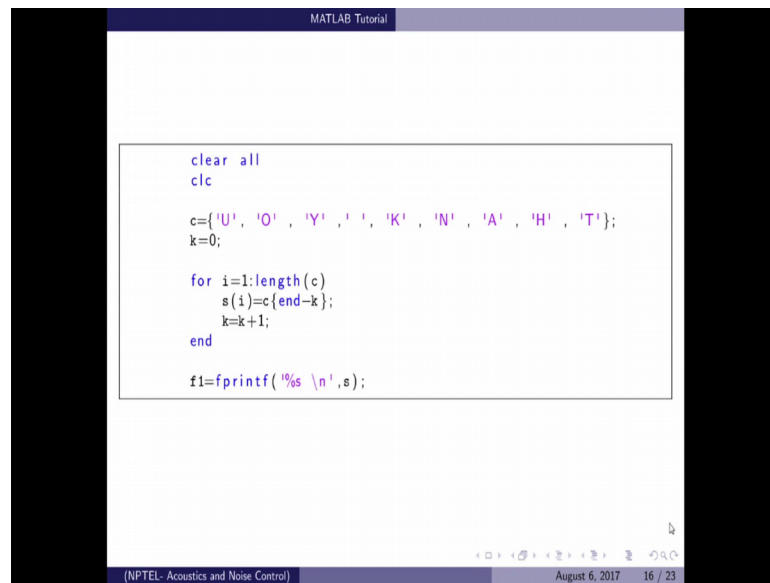
(Refer Slide Time: 51:50)



So, this animation shows the total energy of the body in a projectile motion over the time. So, at each instant the kinetic energy and potential energy some will remain same though, they may vary individually there some will always remain same.

Similarly, you can pick any concept from these 2 subject and try to animate try to attempt all the MATLAB assignments that will be uploaded during the course it will not only develop your coding skills, but it will also help in understanding; what we are learning the different types of waves which will come across can be beautifully visualize using MATLAB animations.

(Refer Slide Time: 52:43)



```
clear all
clc

c={'U', 'O', 'Y', 'I', 'K', 'N', 'A', 'H', 'T'};
k=0;

for i=1:length(c)
    s(i)=c{end-k};
    k=k+1;
end

f1=fprintf('%s \n',s);
```

The image shows a MATLAB tutorial slide with a code editor window. The code defines a cell array 'c' containing the characters 'U', 'O', 'Y', 'I', 'K', 'N', 'A', 'H', 'T'. It then iterates through the array from the end to the beginning, storing each character in a string 's'. Finally, it prints the string 's' using 'fprintf'.

So, we come to the end of tutorials if you have any doubt please ask the queries in the MATLAB category which has been created in the forum, we will try our best to resolve it.

Thank you.