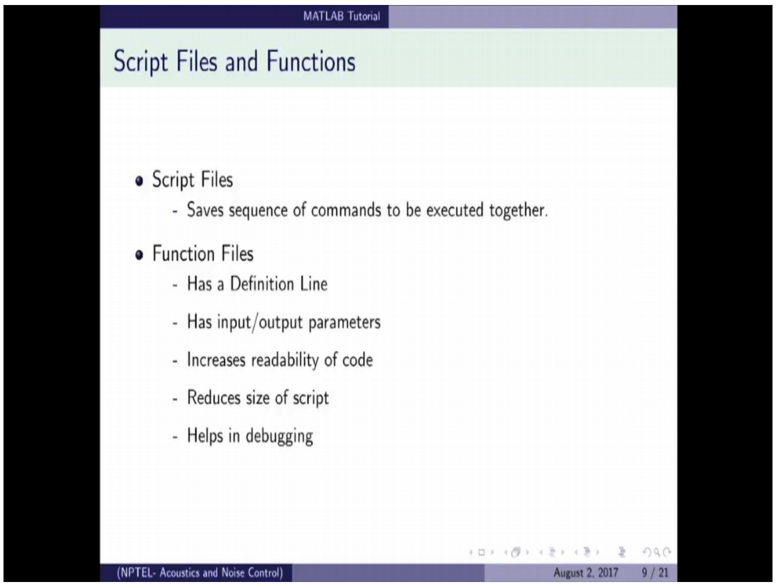
Acoustics & Noise Control

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

Lecture – 40
MATLAB Tutorial – 2

(Refer Slide Time: 00:16)



MATLAB Tutorial

Script Files and Functions

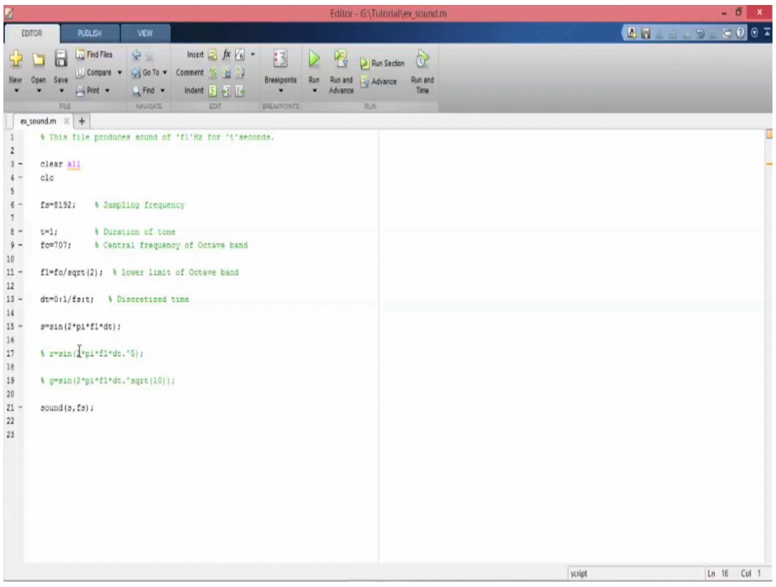
- Script Files
  - Saves sequence of commands to be executed together.
- Function Files
  - Has a Definition Line
  - Has input/output parameters
  - Increases readability of code
  - Reduces size of script
  - Helps in debugging

(NPTEL- Acoustics and Noise Control) August 2, 2017 9 / 21

Hello all and once again welcome to the MATLAB tutorial. This is the second tutorial in the previous one, we studied about the building blocks of MATLAB that is scalars vectors and matrices in this tutorial, we will study about script files function, files plots loops and selection we will also see how we can use all these concepts and create animations in MATLAB. So, let us begin script files are simple dot m files where we write and store a sequence of commands to be executed together instead of writing those commands in the command window we save them in a script file.

So, that we can have access to them whenever we need, we will look into a few script file already saved and some we will right now in dose script files will come across new functions and we will learn about them.

(Refer Slide Time: 01:28)



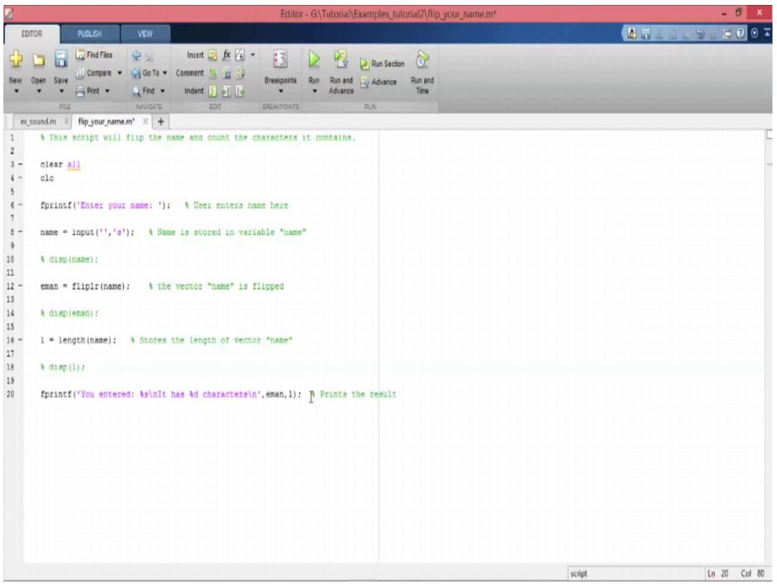
```
% This file produces sound of 'f1'Hz for 't'seconds.

clear all
clc

fs=8192;    % Sampling frequency

t=1;        % Duration of tone
fc=707;     % Central frequency of Octave band

f1=fc/sqrt(2);  % lower limit of Octave band

dt=0:1/fs:t;   % Discretized time

s=sin(2*pi*f1*dt);

% r=sin(2*pi*f1*dt.^5);

% g=sin(2*pi*f1*dt.^sqrt(10));

sound(s,fs);
```

Let us see the example script file sum dot m. So, this file will produce sound at f one hertz for t seconds, it is always nice to write information about the file on top as it will help others to understand the code and the green text which you see here are called comments to write a comment we begin with a percent symbol everything that comes after percent is information for the user MATLAB will ignore all those lines it is a good practices to write comments whenever and wherever it is needed now f s is the sampling frequency which you will use to sample our time t which is the duration of the tone f c is the center frequency of the octave band.

You will learn more about this octave band later in the course f 1 is the lower limit of the octave band and d t will store the discretized time as we with the vector which will store the sin 2 pi f 1 d t and sound s comma f s will produce the sound which is contained in vector s at sampling frequency f s. So, let us hear the sound, observe here that we have define 2 more variables r and g, but they are commented. So, it means that though we have written these statements, MATLAB will not evaluate r and g as you can see here in workspace MATLAB has evaluated only the variable s, but it has ignored the variables r and g which are commented. So, if I want to comment s the shortcut key is control r and if I want to uncomment r the shortcut keys control t.

Suppose I want to evaluate all these s and r and g and I want to play the sounds one after the other. So, I will create a vector called tune and the elements will be vector s vector r and vector g and we will give this input tune to sound. So, we run the code and let us hear.

(Refer Slide Time: 04:28)



```
% This script will flip the name and count the characters it contains.

clear all
clc

fprintf('Enter your name: ');   % User enters name here

name = input('','s');   % Name is stored in variable "name"

% disp(name);

eman = fliplr(name);    % the vector "name" is flipped

% disp(eman);

l = length(name);   % Stores the length of vector "name"

% disp(l);

fprintf('You entered: %s\nIt has %d characters\n',eman,l); % Prints the result
```

Similarly you can play with this tune command and you can generate the various sounds. Now let us write script file transcribe. So, let us write script file such that it will ask the user its name and then it will flip the name and it will tell how many characters are present in that name. So, we will begin with information about our code, this script will flip the name and count the characters it contains, we will clear the work space will clear the command window and now we will come across the new function called f print f.

So, what is f print f does is that it allows us to write information on the screen for the user to view, this is very important when the user interaction is involved. So, let us see; how we can use this command. Now we have to ask the user its name. So, we will write f print f. Enter your name, now when the user will input his name, we should have a mean to store it for that; we will use the command input. So, we will store that in the variable name. So, name will be input and if we put s after that MATLAB will know that what the user as input is a string, now let us first display this much display name. So, we will first save our file let us name it and save now we will run it. So, MATLAB is asking our name and it is waiting for our input. So, let us say we enter and MATLAB has displayed our name.

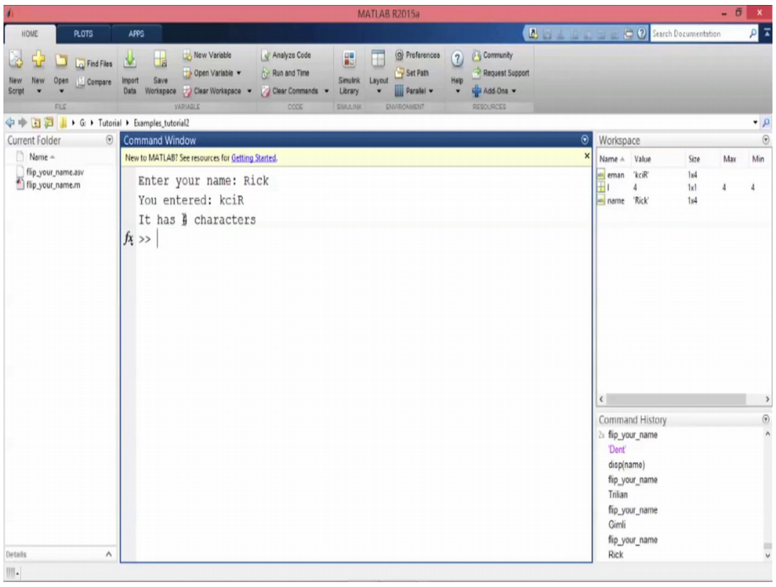
Now, if I want to keep this command, but I do not want MATLAB to execute it, I will do a control r and MATLAB will comment it now what happens if I omit this s if I do not tell MATLAB that the input is a string? So, let us see what happens, if I now run the code again this time a MATLAB has given us an error we wanted to input a string, but to input string we have to enclose the characters in single inverted comma and since we have not done that MATLAB does not know that what we have input is a string and it will give us an error. So, if we have missed this as in input and still if you want to enter string we can do it this way and now we will enclose our name into inverted commas and this time MATLAB will display our vector.

So, now MATLAB has now displayed our vector if we include this as then we need not enclose our name each time in the inverted commas now we have to flip the name. So, we do this by using a command called flip L r; L r means left right flip left and right similarly there is one more command flip u d which stands for flip up down for no particular reason let us say our variable is this and flip L r flip the vector name now let us see what the variable stores we will again run our code and this our name time and the press enter we expect that it will be flipped let us see and it is. So, one part of our code is over that we have flipped the name. Now let us see how to count the characters in it for that we will use command length will give me the length of the vector.

So, let us say we have a vector v which contains 3 rows. So, length v will give me length of vector that is 3. So, we will store it in variable L and length name. Now we will display length and see we get result and we expected to be 5 and yes, Gimli has 5 characters. Now we will see how to present our results in much more better manner again here we will use the command f print f. So, f print f you entered percent s will tell MATLAB that here a string will come backslash n will tell MATLAB to start a new line and then we will tell the user how many characters it has. So, it has percent d percent d will tell MATLAB clear and integer value will come. So, it has percent d characters again a backslash n to start a new line closing inverted comma.

So, that MATLAB will show only the string which you have entered within the inverted commas comma. Now we have to tell MATLAB that what will come in place of percent s and what will come in place of percent d that we tell by sequentially mentioning the variable names. So, the variable e m a n will come in place of percent s and L will come in place of percent d let us write a few comments. So, that we will understand it later user enters name here name is stored in variable name the vector name is flipped towards the length of vector name prints the result. So, let see.

(Refer Slide Time: 13:35)



```
New to MATLAB? See resources for Getting Started.

  Enter your name: Rick
  You entered: kciR
  It has 4 characters
fx >>
```

Workspace

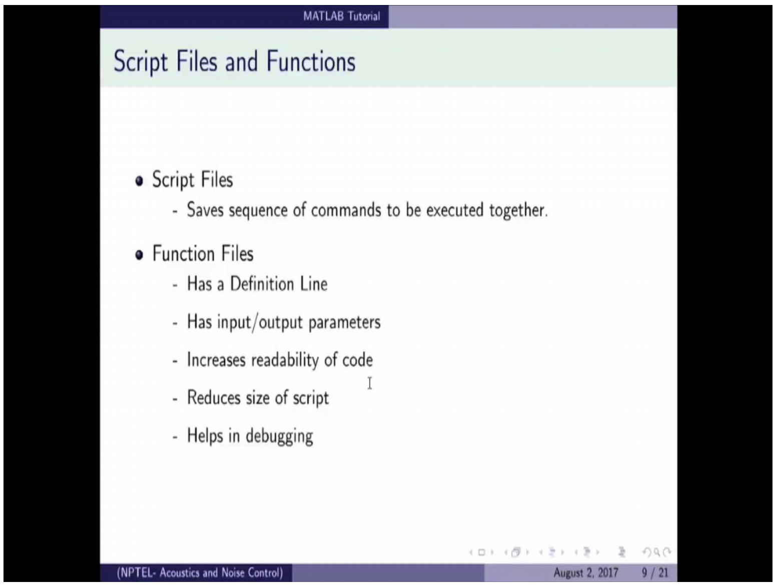
Name	Value	Size	Max	Min
eman	'kciR'	1x4		
l	4	1x1	4	4
name	'Rick'	1x4		

Command History

```
flip_your_name
'Dent'
disp(name)
flip_your_name
Trilian
flip_your_name
Gimli
flip_your_name
Rick
```

So, this is how MATLAB will represent the result, it will flip the name and it will tell me how many characters are present in it.

(Refer Slide Time: 13:53)



MATLAB Tutorial

Script Files and Functions

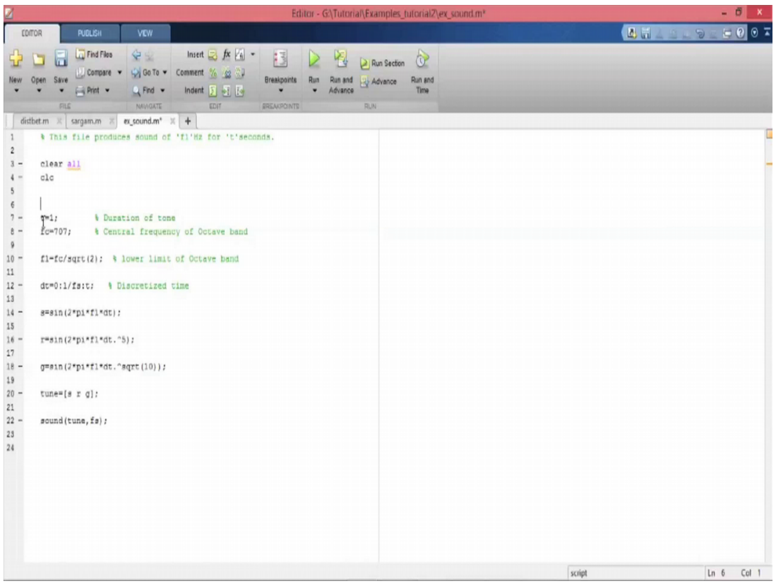
- Script Files
  - Saves sequence of commands to be executed together.
- Function Files
  - Has a Definition Line
  - Has input/output parameters
  - Increases readability of code
  - Reduces size of script
  - Helps in debugging

(NPTEL- Acoustics and Noise Control) August 2, 2017 9 / 21

Now, let us come to function files. The difference between script file and function file is that function file will have a definition line and it will also have something called input output parameters, if you remember in script files, we have to define the variables in the code in order to use them, we could not pass any variables from outside, but in case of function files we can do. So, and this give them a lot of flexibility also the variables that are created in the function are local that is they do not show up in the workspace, they exist within the function only we will see what the passing of variables mean in a minute here are some of the advance stages of using functions in your code, they increase the readability of the code they reduce the size of the script and they also help in debugging

So, let us see a few examples of functions.

(Refer Slide Time: 15:04)



```
% This file produces sound of 'f1'Hz for 't'seconds.

clear all
clc


t=1;          % Duration of tone
fc=707;       % Central frequency of Octave band

f1=fc/sqrt(2);  % lower limit of Octave band

dt=0:1/fs:t;   % Discretized time

s=sin(2*pi*f1*dt);

r=sin(2*pi*f1*dt.^5);

g=sin(2*pi*f1*dt.^sqrt(10));

tune=[s r g];

sound(tune,fs);
```

So, let us re-visit our first script file in which we calculate at the distance between 2 points in that script file if x naught is changing, we have to change x naught every time save the script file and run it though it has advantages over writing those commands in the command window; still is there are too many x naught values, we have to each time we have to visit the script file change x naught values and then find the result again and again. So, let us write a function file and see how it makes our life simpler. So, to start writing a function file we can create it with new and a function first we will rename our function let us call it distbet distance between 2 points the input arguments which we will be passing into this function are x naught y naught x 1 and y 1 the output argument will be the distance between these 2 points and it will be stored in r the same variable which we used in our script.

So, this first line just after the definition line this is the definition line and the first line will tell information about your function. So, if you search in command window using help what you will see is this line. So, whatever function does is it calculates distance between 2 points and some details will go in the second line provide x and y coordinates of 2 points the order has to be x naught y naught x 1 and y 1 and now let us compute r. So, r is square root x naught minus x 1 square plus y naught minus y 1 square.

First let us save this file one thing should be carefully noted is that whatever function name you put here you should save your file with the same function name MATLAB will automatically do it for you, but in case you should remember that this function name should be same as your file name. So, here we save our function now how to use it. So, if you have to use the function first requirement is that it should be present in your current working directory. So, here it is. So, here it is present in this directory. So, we can use it. So, dist between that is how we call the function. So, x naught it says 0 y naught is 0 x 1 is 2 and y naught is also y y 1 is also 2.

And let us say we store that distance in y one more thing observe here is that the output parameter here is r, but we are storing our result in y. So, we can do that it is not absolutely necessary to use the same output parameter as your variable to store its output. So, when we press enter MATLAB will give me the distance between that these 2 points notice one thing here that though we are passing the variables value 0 0 2 2 and they are getting stored in variables x naught y naught x 1 y 1 the variables will not show up in the works space only why which we have defined in the command window has appeared in the workspace. So, this is how we mean that the existence of the variables in function is limited 2 function. So, I cannot use x naught outside, if I try to use MATLAB will say variable x naught does not exist.

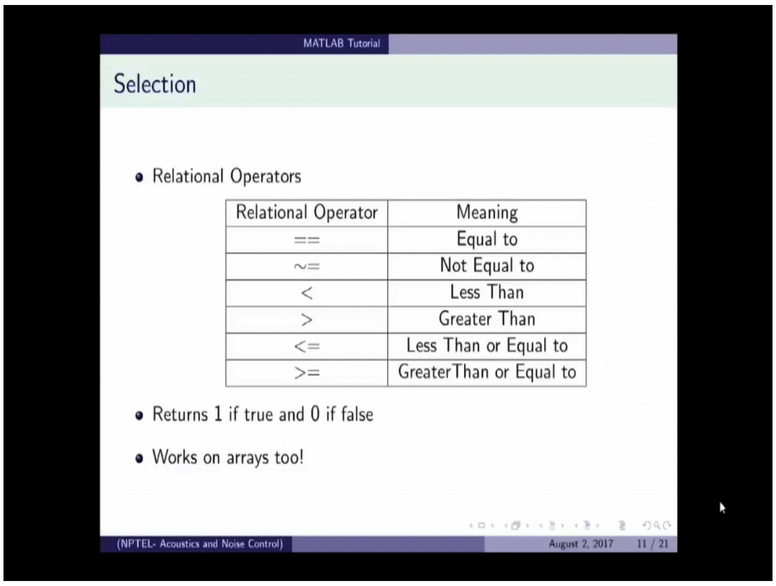
Now, let see another function if you remember our sound script file there to find sounds at few different frequencies we had to write 3 commands. So, instead of that we can just write one function and use it to generate as many sound vectors we need let us create the function let us called our function Sargam the input arguments will be frequency at which the sound will be played and the time the duration of the sound and output parameter the sound information will be contained in y. So, again here goes the function information this function will produce sound at f hertz for t seconds.

Some details user should provide frequency in hertz and time in seconds again f is our sampling frequency will define a vector capital t which is our discretized time begins from 0 sampled at disrupt up to time t which is input from user and y will be sign to pi f and capital t. So, let us save this sargam and now let us revisit our script file now instead of defining all these things here we just have to mention time and the frequency let us say our frequency is 7 0 7 and s or s a will be. So, I will call the function Sargam by its name, I will tell that the frequency is at c as you can see MATLAB is prompting help it will tell us what it expects us the next input.

Now, that we have input the frequency MATLAB will ask me; what is the time. So, time is t 1 second parentheses is close and we have done let us see how it works. Now if you want to say which is at frequency for the time t, similarly we can; let us say we produce a few more vectors and here we have define 4 sounds. So, instead of writing sin 2 pi f t sign 2 pi f t for each of them, we are just passing the frequency and time for each one of it and the function Sargam will evaluate the vector and if the result will be stored in the corresponding variables.

So, let us just create a tune using this and let us hear; I will suppress the output and tune. So, let us hear it you can create many more tunes using it, it is a very good command sound play around with it, there are 2 more function files that we will write, but for that we will need the prerequisite of if else statements and plot functions. So, we will first learn about them and after that we will again write to more function files in much more details with much more complexity.

(Refer Slide Time: 27:01)



MATLAB Tutorial

Selection

- Relational Operators

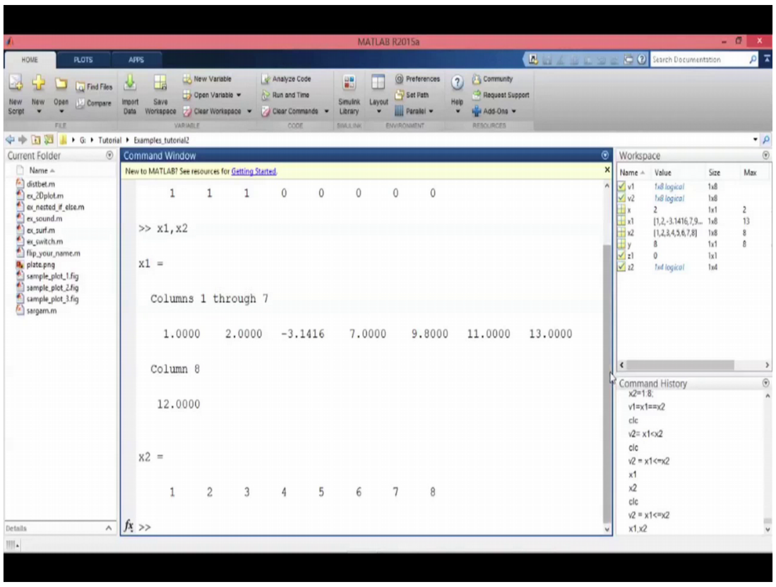
Relational Operator	Meaning
==	Equal to
∼=	Not Equal to
<	Less Than
>	Greater Than
<=	Less Than or Equal to
>=	GreaterThan or Equal to

- Returns 1 if true and 0 if false
- Works on arrays too!

(NPTEL- Acoustics and Noise Control) August 2, 2017 11 / 21

We will now see selection, first let us learn what relational and logical operators are so relational operators as the name suggests compares 2 statements and returns 1 if true and 0 if false. So, this table here lists relational operators let us try a few of them. So, let say.

(Refer Slide Time: 27:27)



```
     1     1     1     0     0     0     0     0

>> x1,x2

x1 =

  Columns 1 through 7

    1.0000    2.0000   -3.1416    7.0000    9.8000   11.0000   13.0000

  Column 8

   12.0000


x2 =

     1     2     3     4     5     6     7     8

fx >>
```

Workspace

Name	Value	Size	Max
v1	1x8 logical	1x8	
v2	1x8 logical	1x8	
x	2	1x1	2
x1	[1,2,-3.1416,7,9...	1x8	13
x2	[1,2,3,4,5,6,7,8]	1x8	8
y	8	1x1	8
z1	0	1x1	
z2	1x4 logical	1x4	

Command History

```
x2=1:8;
v1=x1==x2
clc
v2= x1<x2
clc
v2 = x1<=x2
x1
x2
clc
v2 = x1<=x2
x1,x2
```

X is 2 and y is 8. So, first we will see the equal to equal to operator. So, if I write x equal to equal to y what i m asking MATLAB is to compare if x and y are equal that is if x and y have same value. So, if it is true z 1 should get value 1 if it is false we will get 0, since they are not equal z 1 is 0. Now let us create a vector of 0s and 1s using relational operators. So, we will store that in z 2 and first let us ask MATLAB if x is less than y, next we will ask MATLAB if x is greater than y, then we may ask is x plus 5 greater than or equal to y minus 2 and if let us say, this x square less than or equal to or let say if x cube is less than or equal to y square.

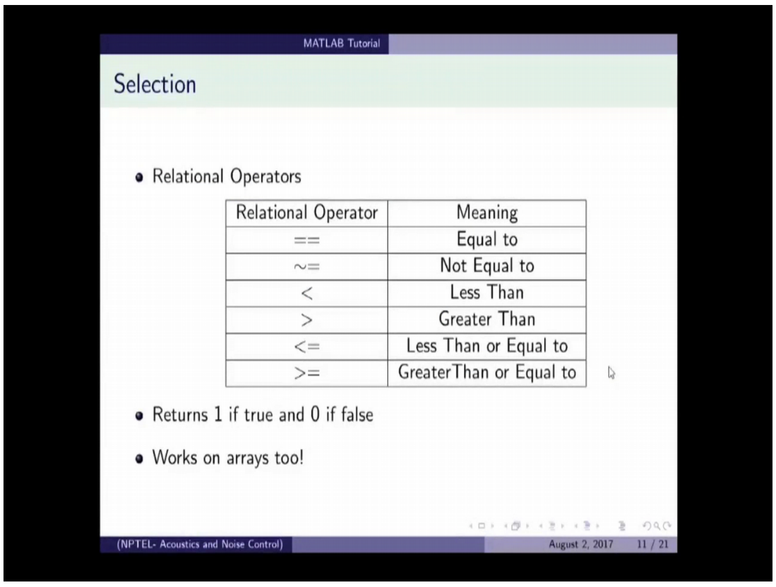
So, we will accordingly get a vector of 0s and 1s. So, here we see the entries of z 2 are 1 0 1 1. So, is x less than y yes. So, it is true is x greater than y no it is a false we get second entry as 0. Now x plus 5 is 7 and y minus 2 is 6. So, is 7 greater than or equal to 6, yes. So, it is a true statement and we get one, similarly is x cube less than or equal to y square. So, is 8 less than or equal to 64 definitely therefore, the statement is true and again we get one. So, that is how the relational operators work.

Now, we will see how we can use them with array also. So, suppose I have a vector x one which is let us say 1 2 minus 5 and let say 79.8 11 13 12 here and we have another vector 1 to 8 and we use semi colon to surprise the output. Now if you want to compare the elements if you want compare 2 arrays. So, if I say, let us for the result in v 1 if I say v 1 equal to x 1 equal to equal to x 2. So, MATLAB will compare each element of vector x 1 with each element of vector x 2, it will see if they are same.

If they are equal MATLAB will return one or else it will return 0. So, as we can see only the first 2 elements will match rest since its false we are getting a 0. Now let us create a vector v 2 which will store the output of this relation will compare elements of x 1 are they less than or equal to elements of x 2. So, this is what we get. So, this is our x 1 and x 2. So, as you can see here only the first 3 elements in x 1 are either less than or equal to elements in x 2.

From fourth element onwards, the statement is false and all the elements in v 2 are 0.

(Refer Slide Time: 32:10)



MATLAB Tutorial

Selection

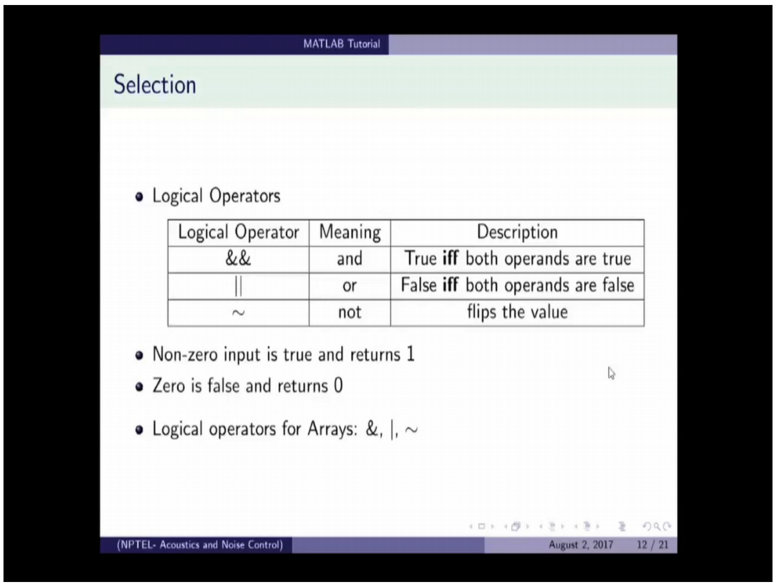
- Relational Operators

Relational Operator	Meaning
==	Equal to
~=	Not Equal to
<	Less Than
>	Greater Than
<=	Less Than or Equal to
>=	GreaterThan or Equal to

- Returns 1 if true and 0 if false
- Works on arrays too!

(NPTEL- Acoustics and Noise Control) August 2, 2017 11 / 21

(Refer Slide Time: 32:14)



MATLAB Tutorial

Selection

- Logical Operators

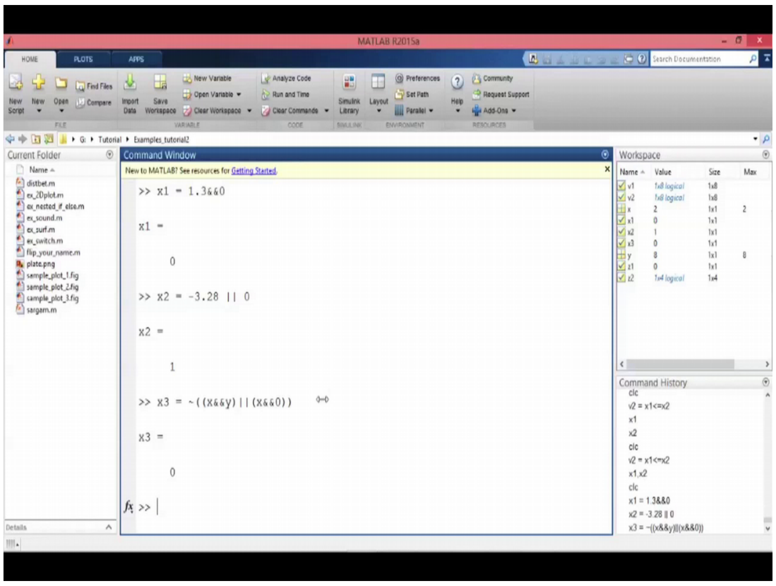
Logical Operator	Meaning	Description
&&	and	True **iff** both operands are true
||	or	False **iff** both operands are false
~	not	flips the value

- Non-zero input is true and returns 1
- Zero is false and returns 0
- Logical operators for Arrays: &, \|, ~

(NPTEL- Acoustics and Noise Control) August 2, 2017 12 / 21

Now let us see logical operators, so logical operators determine the true or false state using Boolean algebra. In Boolean algebra 1 is true and 0 is false, but in MATLAB in the non 0 value is true and 0 is false, here also the logical operators r and r naught, the symbols are tabulated in this table in the truth tables for the operators are same as that of Boolean algebra. So, now, let us see few examples again end will return one if and only if both the statements are true.

(Refer Slide Time: 32:53)



New to MATLAB? See resources for Getting Started.

```
>> x1 = 1.3&&0

x1 =

     0

>> x2 = -3.28 || 0

x2 =

     1

>> x3 = ~((x&&y)||(x&&0))

x3 =

     0

>> 
```

Workspace

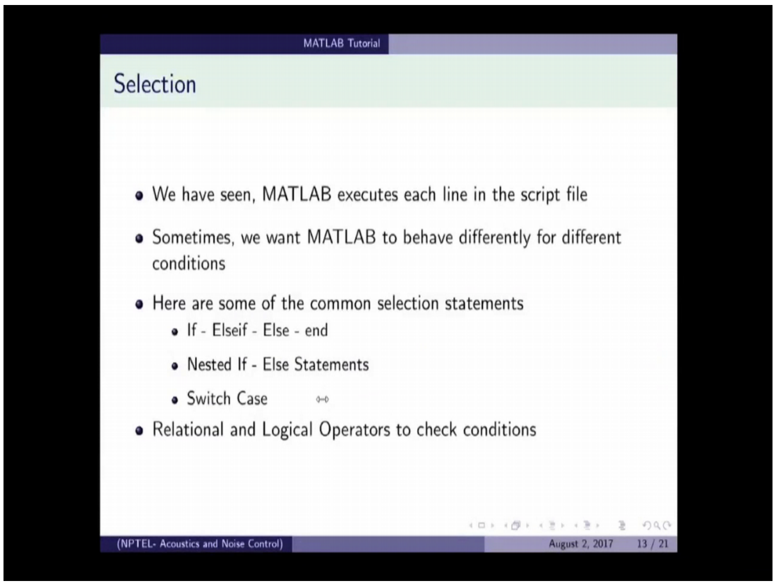
Name	Value	Size	Max
v1	1x8 logical	1x8	
v2	1x8 logical	1x8	
x	2	1x1	2
x1	0	1x1	
x2	1	1x1	
x3	0	1x1	
y	8	1x1	8
z1	0	1x1	
z2	1x4 logical	1x4	

Command History

```
clc
v2 = x1<=x2
x1
x2
clc
v2 = x1<=x2
x1,x2
clc
x1 = 1.3&&0
x2 = -3.28 || 0
x3 = ~((x&&y)||(x&&0))
```

So, true and true will be true for and operator and in MATLAB any non 0 number is true therefore, let us see what x 1 will be if I say 1.3 and 0. So, it is true and false. So, we expect x 1 to be 0 and it is now, similarly we can check for or so, x 2 will be let us say minus 3.28 or 0. So, or will be false if and only if both the operands are false. So, in this case since it is true or false the solution will be true. So, x 2 will store value one what will be the output of this MATLAB will first evaluate the inner brackets. So, first it will evaluate x and y since x and y are both non 0 numbers this will this is actually true and true which will result in true and this bracket is true and false which will result in false. So, true or false will again result in true and then we are using a not operator. So, not true will be false. So, x 3 should be having value 0. So, this is how we use the relational operators and the logical operators.

(Refer Slide Time: 34:53)


MATLAB Tutorial

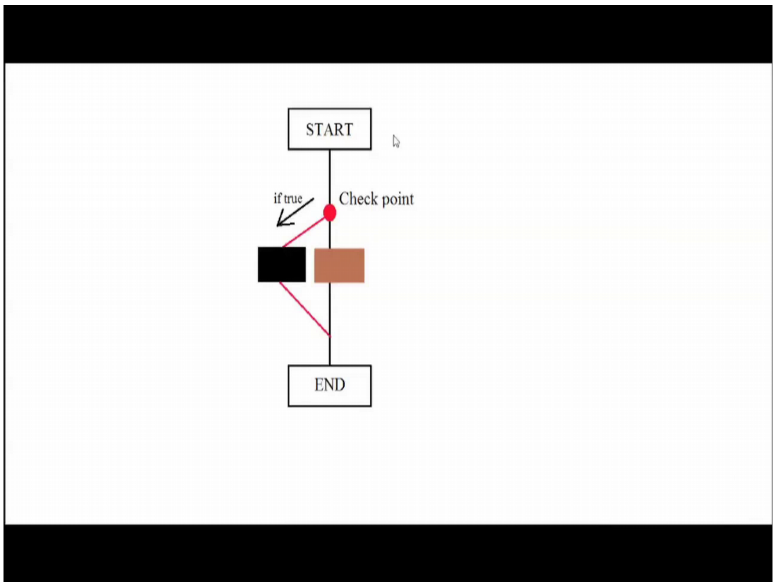
Selection

- We have seen, MATLAB executes each line in the script file
- Sometimes, we want MATLAB to behave differently for different conditions
- Here are some of the common selection statements
  - If - Elseif - Else - end
  - Nested If - Else Statements
  - Switch Case
- Relational and Logical Operators to check conditions

(NPTEL- Acoustics and Noise Control) August 2, 2017 13 / 21

Now, let us see the actual selection part. So, we have previously seen that MATLAB executes statements sequentially one after other not always we will want MATLAB to run the entire code as it is, we may want to run different statements under different conditions. So, we will accomplish this using the selection statements MATLAB provides 2 options for selection first is else if else statement other is the switch statement. So, how does MATLAB decide which case to run it uses the relational and logical operators to arrive at the decision.

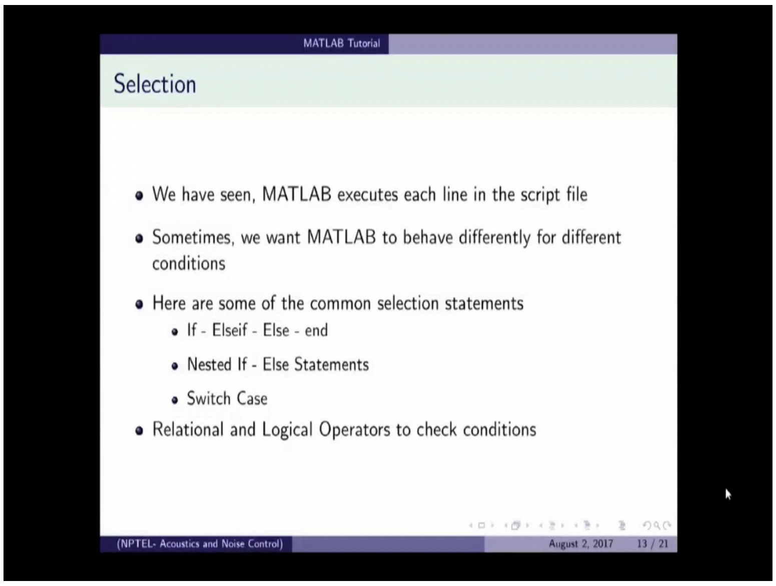
(Refer Slide Time: 35:42)



If we have to represent this selection in form of a block diagram, we can illustrate it as shown here.

So, this is the start of our program then we will go to this check point where it will check a certain condition using a relational operators or logical operators. For example, we have checked that whether x is less than y something like that. So, if x is less than y you would want this block to run. So, if it is true MATLAB will run execute this statements and again will join the main program and we will end.

(Refer Slide Time: 36:39)



MATLAB Tutorial

Selection

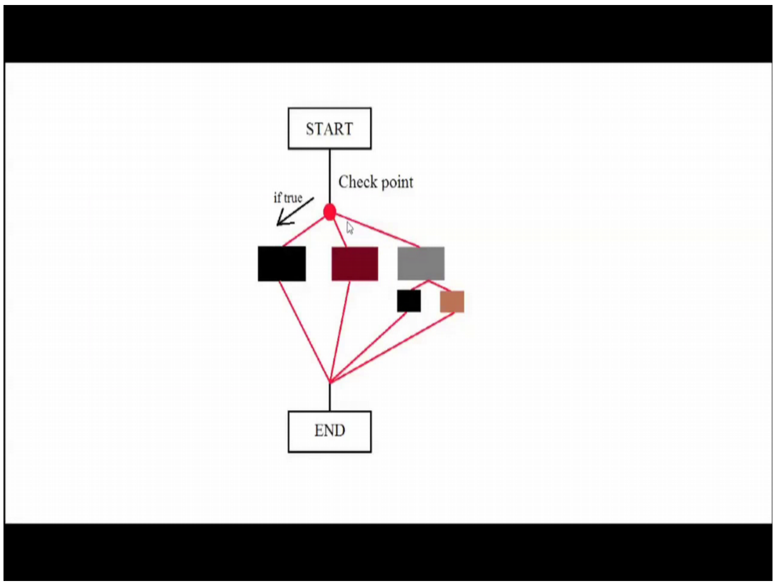
- We have seen, MATLAB executes each line in the script file
- Sometimes, we want MATLAB to behave differently for different conditions
- Here are some of the common selection statements
  - If - Elseif - Else - end
  - Nested If - Else Statements
  - Switch Case
- Relational and Logical Operators to check conditions

(NPTEL- Acoustics and Noise Control) August 2, 2017 13 / 21

If it is false MATLAB will execute these statements and again you will join main program and we will end many a times, there will not be just 2 conditions like if it is true follow this or follow that we may have several options like if it is true follow one path else there may again be 2 options.

So, in that case we can use this nested if else statements or this if else if else statements. So, again we will see how it is represented.

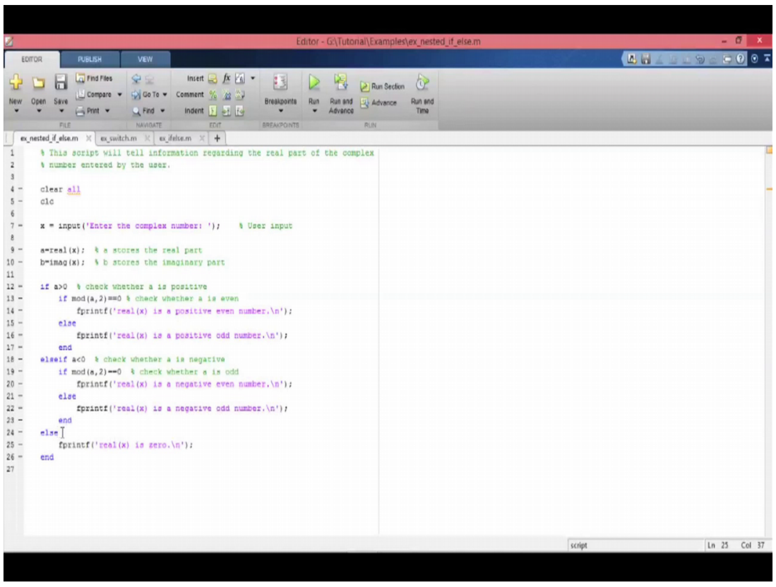
(Refer Slide Time: 37:06)



So, here again we start our program we reach a checkpoint, now let us say here we check that the condition if x is equal to y that is if x and y have both same values if it is true MATLAB will execute these commands and again we will follow the main program and terminate it suppose x is not equal to y. So, in that case we may have again 2 options whether x is greater than y or x is less than y. So, again if x is less than y we may follow this path if x is greater than y we will follow this path. So, it will execute these commands and again here we may have some nested paths.

So, this is the main if else statement then here we are again have an; if else statement within this if else statement. So, here a again let us say MATLAB will check whether x is positive or negative. So, if x is positive we will execute these commands if the x is negative we will execute these commands and then we will return to the main program. So, in this way you can use the nested if else statements and you can explore as many conditions or choices available. So, let us see example of a simple if else statement then we will see example of a nested if else statement.

(Refer Slide Time: 39:01)



```
% This script will tell information regarding the real part of the complex
% number entered by the user.

clear all
clc

x = input('Enter the complex number: ');    % User input

a=real(x);  % a stores the real part
b=imag(x);  % b stores the imaginary part

if a>0  % check whether a is positive
    if mod(a,2)==0 % check whether a is even
        fprintf('real(x) is a positive even number.\n');
    else
        fprintf('real(x) is a positive odd number.\n');
    end
elseif a<0  % check whether a is negative
    if mod(a,2)==0  % check whether a is odd
        fprintf('real(x) is a negative even number.\n');
    else
        fprintf('real(x) is a negative odd number.\n');
    end
else
    fprintf('real(x) is zero.\n');
end
```

Let us write it down from beginning first we will tell; what the script does. So, this script will tell if the entered number is real or complex. So, first we will clear the command window. So, first we will clear the workspace then will clear the command window now we will use the function input which we had seen in the script files. So, n will store the number and you can write input and instead of using print f and input combination we can directly write the prompt in these inverted commas.

So, we will ask user to enter number and we will wait for user input. Now since we are expecting integers or floats that is we are not expecting the user to input a string. So, we will not use that comma as which we had previously used in script. So, if we use it here then MATLAB will assume that we have to enter a string, but here, we do not want string we want numbers. So, we will not use it and let us leave it at that.

Now, i m variable will store the imaginary part of the number which the user will input. So, how do we extract the imaginary part of a number we have, we can use the command i m a g. So, i m a g will extract the imaginary part from the number. Now we can use if else statement. So, we start with if we compare i m to 0. So, if imaginary part is 0 then print and new line else. Now here we need not mention the condition that i m is not equal to 0 because if i m is equal to 0 the other condition is i m not equal to 0. So, if this is false this has to be true and do not forget the end to tell MATLAB that here the; if else statement ends. So, let us save our program and let us run it.

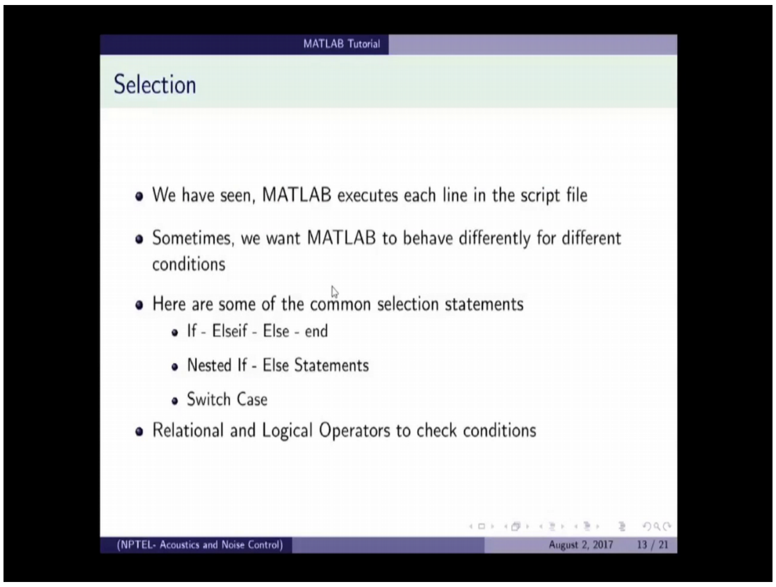
This time we will run it using the name of the file x underscore if else. So, we will go the command window. So, MATLAB has asked us the number. So, how do we input a complex number? So, we will enter the real part 2 and let us say plus now we have input the imaginary part. So, in MATLAB i and j represent imaginary numbers. So, generally I prefer to write it as 1 j star 3. So, here the imaginary part is 3 and let us say what MATLAB gives us. So, it tells us that n is a complex number. Now let us run the program with 3 plus 0. So, it tells that mat n is a real number. So, what happens if I just enter 3? So, MATLAB will again say that n is real number because the imaginary part of 3 is 0 now it may happen that sometimes, we miss this double equal to and just write single equal to.

Now, let us see what happens MATLAB will immediately tell you that this might be an invalid syntax, but let us ignore it for a moment and see what happens. So, MATLAB does not run the file and it will tell us that the equal to sign, if it is used single, it means that we are assigning the value to the right of it into the variable which is on the left. So, in order to compare we have to use 2 equal to signs. So, this thing should be kept in mind let us write few comments now let us see an example which is nested if else. So, the script will tell information regarding the real part of the complex number which is entered by the user.

So, here again we will ask user to enter a complex number. So, here we will explicitly ask user to enter a complex number and x will store it. Now to extract the real part we will use the command real parenthesis complex number and we have seen how to extract imaginary part. So, a will store the real part and b store the imaginary part now we will first we check if a is positive or negative. So, first we check here if a is greater than 0. Now if a is greater than 0, then again we will check 2 more conditions that is x a even or is a an odd number. So, if mode of a v 2 is 0 that is if a is divisible by 2, we know that a is an even number. So, MATLAB will ask MATLAB to print that the real part of x is a positive even number. So, and if mod of a comma 2 is not equal to 0 which means that a is an odd number and MATLAB will print that it is a positive odd number.

So, what happens if this condition fails? So, if a is not greater than 0, it may mean that either a is less than 0 or a is equal to 0. So, we say if this fails is, if a is less than 0, then we again check whether a is odd or even accordingly we will print the statement. Finally, if a is neither greater than 0 and a is nor less than 0, there is only one possibility a is equal to 0. So, MATLAB will tell us that a is 0 the real part of x is 0. So, let us run this program. So, we will enter a complex number 2 minus 3 j or 1 j into 3. So, our real part of this number is 2 which is positive and even number. So, let us see one more example. So, now, let us see an odd number a negative odd minus 3 plus 8. So, now, MATLAB should tell me that my real number is a negative odd and it is now; let us enter 0 plus or you can just say 1 j into 7. So, MATLAB will tell the real part of x is 0.

(Refer Slide Time: 48:18)



MATLAB Tutorial

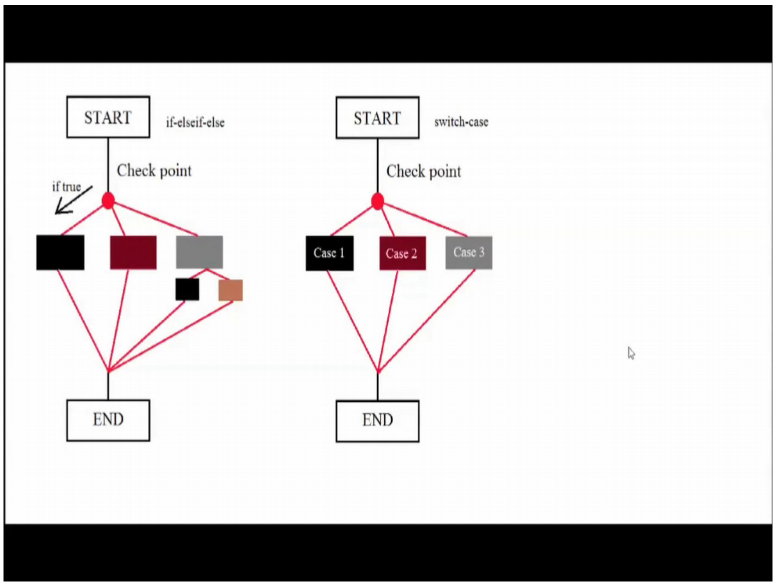
Selection

- We have seen, MATLAB executes each line in the script file
- Sometimes, we want MATLAB to behave differently for different conditions
- Here are some of the common selection statements
    - If - Elseif - Else - end
    - Nested If - Else Statements
    - Switch Case
- Relational and Logical Operators to check conditions

(NPTEL- Acoustics and Noise Control) August 2, 2017 13 / 21

We can also use this switch statement instead of nested if else statement; note that we cannot use the relational operators like less than or greater than in switch statement we can only check whether the condition is equal to several possible cases. So, let us see the block diagram of switch case.

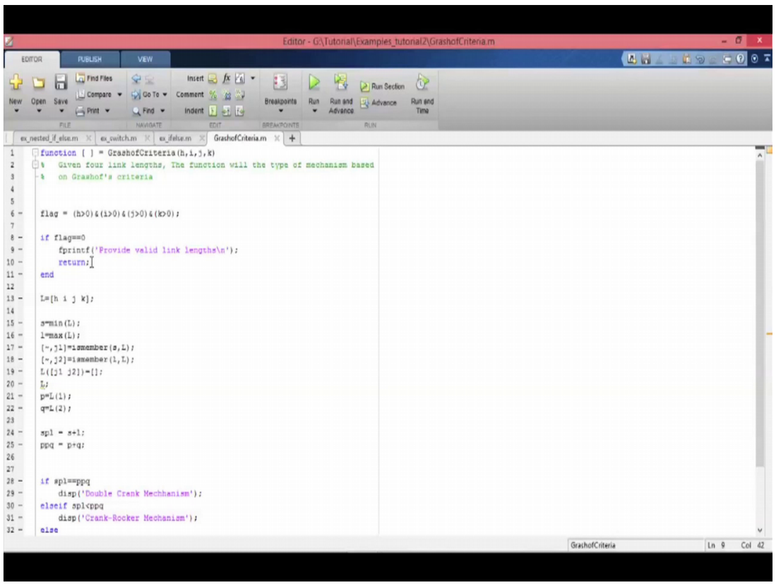
(Refer Slide Time: 48:45)



Again you will start our program and when you will reach checkpoint unlike in case of if else where we could check whether x is less than y or x is greater than y, something like that we cannot do that in switch case statement we have to check whether the case is either case one or case 2 or case 3 if it is equal to either one of these, then MATLAB will execute this particular case or if it is equal to case 2, it will execute this particular case and so on.

So, if we had to replace the switch statement with if else we would have to use only the equal to equal to operators, we could not use other relational operators. So, let us see an example of a switch statement.

(Refer Slide Time: 49:45)



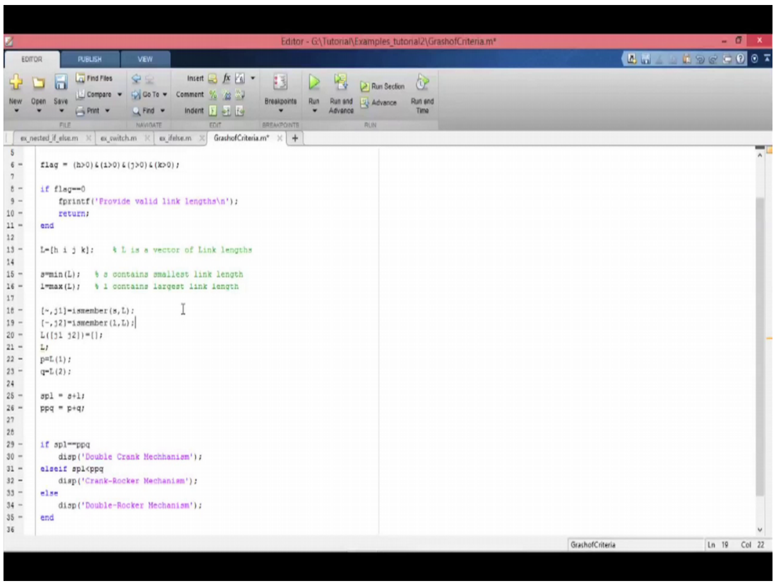
```
function [ ] = GrashofCriteria(h,i,j,k)
%   Given four link lengths, The function will the type of mechanism based
%   on Grashof's criteria


flag = (h>0)&(i>0)&(j>0)&(k>0);

if flag==0
    fprintf('Provide valid link lengths\n');
    return;
end

L=[h i j k];

s=min(L);
l=max(L);
[~,j1]=ismember(s,L);
[~,j2]=ismember(l,L);
L([j1 j2])=[];
L;
p=L(1);
q=L(2);

spl = s+l;
ppq = p+q;


if spl==ppq
    disp('Double Crank Mechhanism');
elseif spl<ppq
    disp('Crank-Rocker Mechanism');
else
```

So, here is an example let us run and see various cases and then we will see; how we have written it. So, the question is do you know the answer if we say yes we know the answer MATLAB will prompt. So, what is the question again let us run the program? So, now, if we say no I do not know the answer MATLAB will ask us not to panic and we will tell that answer is 42, suppose I am kind of rebel and I will not answer in yes or no if I say c. So, in that case MATLAB will tell me please enter yes or no other characters are allowed. So, how did we do this? So, let us see the program. So, again c will store the choice; now here we are using MATLAB to input x string.

So, here again we will use that s formatting operation with input. So, that the user will not have to enclose input in inverted commas then we will ask MATLAB to check whether c is equal to case y or case n. So, if it is equal to case y MATLAB will run this code if it is equal to n MATLAB will run this code and if it is not equal either to y nor n we will use this otherwise block which is an optional block and this will run if none of the cases are matching with c. So, it will print please enter yes or no. So, that is how we can use the switch command. So, here we have seen how to use relational operators to come to a logical decision which code to run in which condition we have seen how to use nested if else statements you have seen how to use the switch statement.

So, let us see this function grash of criterion, it will tell us the type of mechanism given the 4 link lengths I have already written a code for it we will go through it line by line and try to understand it. So, our first task is to check if the link lengths provided by the user are valid or not we will check that using relational and logical operators. So, this flag will store 0 if any one of these conditions fail since; we have used and operator flag will be one if and only if all these conditions are true now suppose the user has provided invalid link length due to which the flag is 0. So, since this condition is true MATLAB will run this if block. So, MATLAB will print a message saying that please provide valid link lengths and then it will execute this command return what it does is that it will exit the function and pass the control outside the function. So, let us see an example for this we will provide some invalid link lengths.

(Refer Slide Time: 53:.32)



```

flag = (h>0)&(i>0)&(j>0)&(k>0);

if flag==0
    fprintf('Provide valid link lengths\n');
    return;
end

L=[h i j k];    % L is a vector of Link lengths

s=min(L);   % s contains smallest link length
l=max(L);   % l contains largest link length

[~,j1]=ismember(s,L);
[~,j2]=ismember(l,L);
L([j1 j2])=[];
L;
p=L(1);
q=L(2);

spl = s+l;
ppq = p+q;


if spl==ppq
    disp('Double Crank Mechhanism');
elseif spl<ppq
    disp('Crank-Rocker Mechanism');
else
    disp('Double-Rocker Mechanism');
end

```

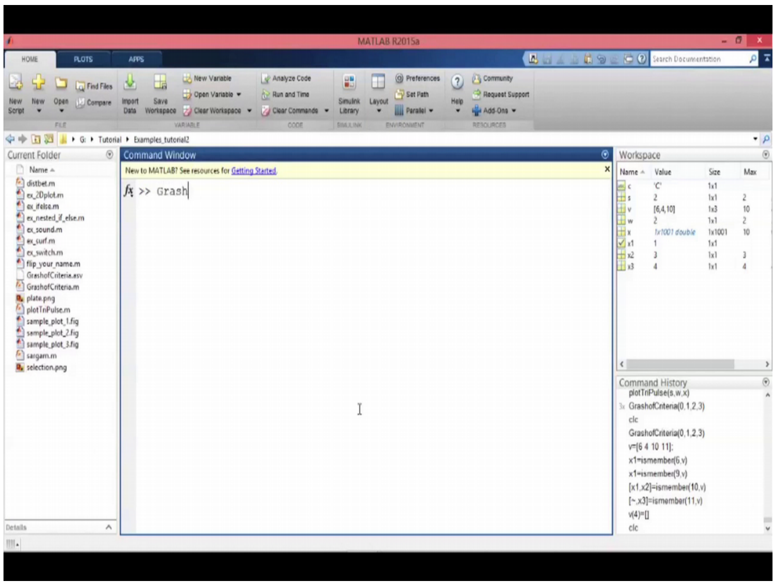
Let say first link length and provide 0, 1. 2 and 3. So, if I press enter now MATLAB will tell me provide valid link lengths and see it has exited the function MATLAB will not execute any of this part it will directly exit the function with this command suppose the user is nice and we are provided with valid link lengths. So, we will make a vector L which says links and the vector will have elements which are the link lengths provided by the user. Now we will store the smallest link length in variable s and the max the largest link length in the variable L to extract the smallest number in the vector, we will use the function minimum that that is m i n of that vector and to extract the maximum of the largest number in the vector, we will use the function max.

So, here let us write comments. So, what we will do now is that we will separate the smallest and the largest link lengths from the other 2 link lengths which we will call p and q. So, how we will do is we will delete the smallest link length and the largest link length from the vector L. So, that whatever remains in vector L will be p and q. So, how we do it is following way first we will see; what is the position of the smallest link length and the largest link length for that; we will use this command is member. So, let us see in command window how this is member works suppose I have a vector v which is one let us say 6, 4, 10 and 11.

Now, if I say x one is member 6 v. So, what MATLAB will check is that MATLAB will check is 6, a member in vector v, if it is member in the vector v matter will return 1. Now if I say 9 is 9 member of vector v no. So, MATLAB will return 0. Now if I want the position of 6 in the vector v, here MATLAB is just saying 1 if it is true and 0 if it is false. Now I want to see where exactly; is this element 6 in the vector v. So, in that case, I will ask MATLAB x 1 comma x 2 enclosed in square brackets equal to is member, let us say 10-10 in v. So, since 10 is a member in v x 1 will store a value 1 and x 2 will store its position which is 3.

So, as you can see here x 1 is 1 and x 2 is 3. Now suppose I just want to work with this variable x 2, I am just interested in the position of the element, I am sure that my vector contains this element, I just want to know the position of it and I do not want to create an extra variable in my code. So, for that I will tell MATLAB I do not want this position or this variable to be created, I just want the position of 11 in my vector v MATLAB will return me 4, we have applied same logic in our code here we are Asking MATLAB to return the position of smallest element in L and j 2 will store the position of the largest element in L which corresponds to the smallest link and the largest link.

(Refer Slide Time: 58:14)



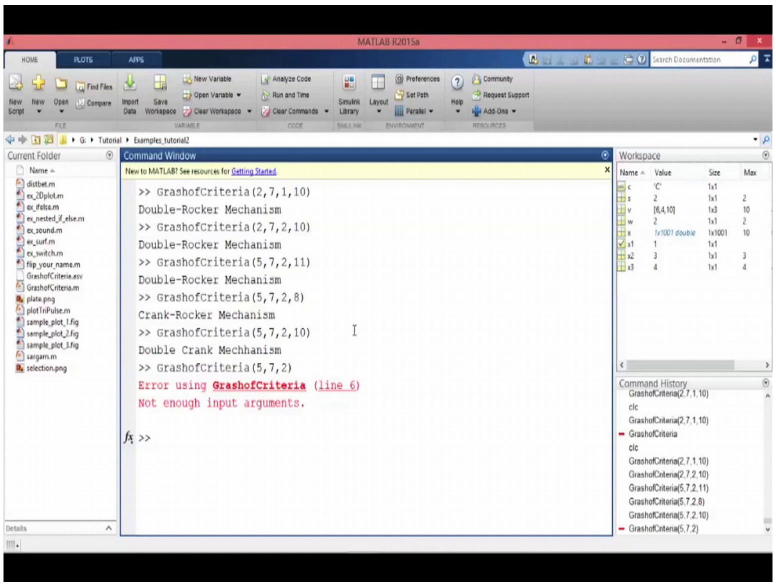
```
>> Grash
```

Name	Value	Size	Max
c	'C'	1x1	
s	2	1x1	2
v	[6,4,10]	1x3	10
w	2	1x1	2
x	1x1001 double	1x1001	10
x1	1	1x1	
x2	3	1x1	3
x3	4	1x1	4

```
plotTriPulse(s,w,x)
3x GrashofCriteria(0,1,2,3)
clc
GrashofCriteria(0,1,2,3)
v=[6 4 10 11];
x1=ismember(6,v)
x1=ismember(9,v)
[x1,x2]=ismember(10,v)
[~,x3]=ismember(11,v)
v(4)=[]
clc
```

So, now we have seen that how to delete entries in the matrix or a vector. So, now, we will delete these 2 elements let us again see in our vector v suppose I want to delete this element 11 from vector v. Now I know from this command that 11 sits at position 4 in vector v. So, what I will do is v of 4 I will set it to empty square brackets. Now if I press enter entry number 4 is deleted. Similarly here I will delete entries in the position j 1 and j 2 which corresponds to the smallest link length and the largest link length. So, that L will contain only link lengths p and q.

(Refer Slide Time: 59:39)



```
>> GrashofCriteria(2,7,1,10)
Double-Rocker Mechanism
>> GrashofCriteria(2,7,2,10)
Double-Rocker Mechanism
>> GrashofCriteria(5,7,2,11)
Double-Rocker Mechanism
>> GrashofCriteria(5,7,2,8)
Crank-Rocker Mechanism
>> GrashofCriteria(5,7,2,10)
Double Crank Mechhanism
>> GrashofCriteria(5,7,2)
Error using GrashofCriteria (line 6)
Not enough input arguments.

>>
```

Name	Value	Size	Max
c	'C'	1x1	
s	2	1x1	2
v	[6,4,10]	1x3	10
w	2	1x1	2
x	1x1001 double	1x1001	10
x1	1	1x1	
x2	3	1x1	3
x3	4	1x1	4

```
GrashofCriteria(2,7,1,10)
clc
GrashofCriteria(2,7,1,10)
GrashofCriteria
clc
GrashofCriteria(2,7,1,10)
GrashofCriteria(2,7,2,10)
GrashofCriteria(5,7,2,11)
GrashofCriteria(5,7,2,8)
GrashofCriteria(5,7,2,10)
GrashofCriteria(5,7,2)
```

So, let us run our code up to this point, so what I will do I will select these remaining and I will do control r and combined them and save the program grash of criteria, we will provide some valid link lengths this time 2 comma 7 comma 1 and 10. So, we can predict few things here that my smallest link length will be one my largest link length will be 10, my j 1 should have value 3 and j 2 should have value 4 and final L after manipulation should contain elements 2 and 7 only. So, now, you can see here L is the vector created with link lengths as is one which is smallest link length L is largest link length j 1 is position of the smallest link length which is s and it is 3.

J 2 is position of the largest link length which is 4 we delete entries corresponding to j 1 and j 2 in vector l. So, that finally, L is remaining with link length 2 and 7 which we have defined as p and q so far. So, good now what we will do is we will assign p as the first element of vector L and q to be the second element of vector L am confidently writing here 1 and 2 because even if j 1 and j 2 are 1 and 2, after we delete those elements from L whatever remains in L are element positions 1 and 2, then we will define 2 more variables s p L which means s plus L which is some of largest and smallest link length and p plus q is some of remaining 2.

So, now we come to the actual grash of criteria which states that if s plus L is equal to p plus q we will get double crank mechanism if it is less than p plus q, we will get a crank rocker mechanism and if it is greater than p plus q we will have a double rocker mechanism. So, this completes our code and let us run entire code once again. So, let us see what kind of mechanism we get with these numbers. So, it is a double rocker mechanism as you can see some of the smallest and the largest is greater than some of the other 2 link lines, now let us see; in this case what we will get is again a double rocker let us make it 8.

So, here we get a crank rocker mechanism because the sum of the smallest and largest is less than some of the remaining 2 and finally, we will see an example of double crank mechanism there the some of the smallest and largest is equal to sum of the other 2 link lengths, sometime it may happen that we are in a hurry and instead of giving 4 link lengths we may pass only 3. So, in that case MATLAB will surely give us an error saying that not enough input arguments, but we do not want to disappoint our user. So, we will use something called as Nargin, what it does is that it will tell me how many input arguments have been provided by the user in this case, since we have provided only 3 the Nargin will be 3.

In this case we have provided all the four. So, the Nargin will be 4 if it is 2 the Nargin will be 2 and so on. So, we can actually use a switch statement in this case. So, let us write switch Nargin which stands for number of arguments in it tells us, how many inputs the user has provided and case let us 3 let us say 3 if user provides 3 arguments then we will tell user a warning we can use this function warning we will tell user that you have provided in sufficient inputs assuming the link length equal to the first input and then we will set our k to be equal to h.

So, what happens is that if I provide only 3 link lengths MATLAB will tell me that you have provided insufficient inputs I needed 4 you provided 3. So, what I will do is I will assume the last input to be equal to the first input. Similarly, if I provide only 2 inputs then again we will print this error message in this case you will assume the link lengths to be equal to the first input. So, in this case our j will be equal to h and k will be also equal to h, it may happen that user has provided only 1 link length. So, again we will tell user you have provided insufficient inputs and in this case, I will be equal to h j will be equal to h and k will also be equal to h and let us end it here. So, let us check our code with this setting now we will provide only 2 link lengths. So, let us say grash of 5 and seven. So, MATLAB will give me a warning that insufficient inputs and assuming the link lengths equal to the first input.

So, my other 2 link lengths are 5 and 5. So, my smallest link length is 5 largest link length is 7 and the remaining 2 link lengths are 5 and 5. So, the sum of the smallest and the largest link length is greater than the sum of the remaining 2. So, we get a double rocker mechanism similarly you can try with someone input or 3 inputs. So, this finishes our function here you can also write some new function take some concept from mechanics or something and try to write it.