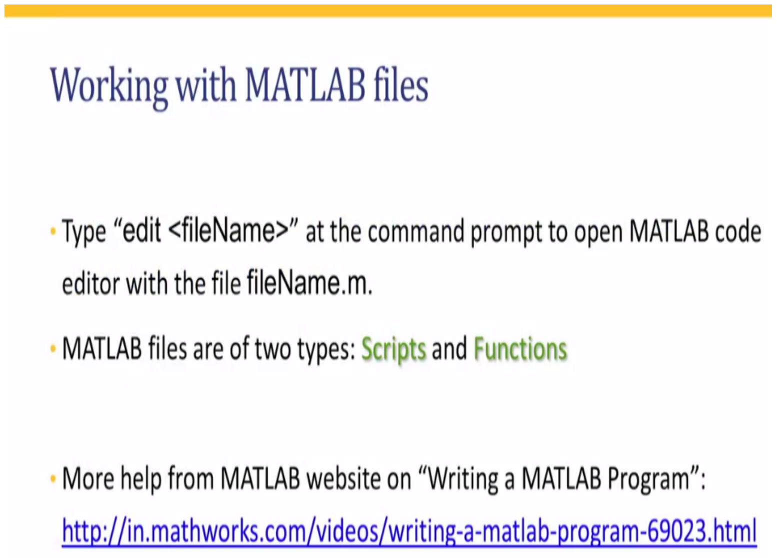
MATLAB Programming for Numerical Computation

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

Module No. #01
Lecture No. #1.4
Introduction to MATLAB programming – Working with files scripts and functions

Hello and welcome to MATLAB programming for numerical computations course. We are in module 1; introduction to MATLAB programming. This is lecture 1-4 where we are going to cover using files in MATLAB. So, MATLAB files come in 2 types' scripts and functions and that is what we are going to cover in this particular lecture okay.

(Refer Slide Time: 00:33)



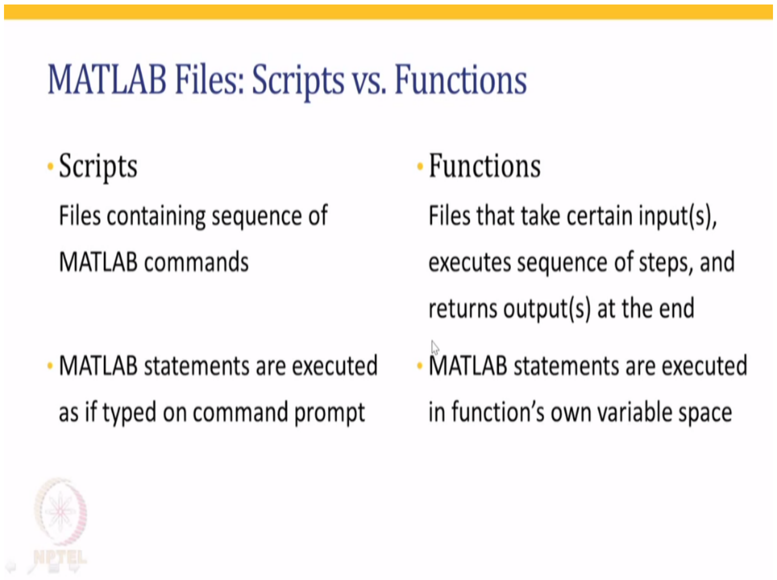
Working with MATLAB files

- Type "edit <fileName>" at the command prompt to open MATLAB code editor with the file fileName.m.
- MATLAB files are of two types: Scripts and Functions
- More help from MATLAB website on "Writing a MATLAB Program": http://in.mathworks.com/videos/writing-a-matlab-program-69023.html

We have done this already a few times in the, in the previous lecture. In order to start editor in MATLAB, we need to type edit followed by the file name. When we do that the editor opens a file called file name .m, .m is the extension to say state that these are MATLAB code file. MATLAB code files are return in ASCII. So they are basically readable by any text editor.

MATLAB provides an excellent editor which is what you should use for typing any MATLAB file, indeed you can use say notepad or any other editor as well which is something that I will not recommend you to do okay. So do use MATLAB editor as we have done before in the previous lectures. For more help you can go to the MATLAB's own website for writing a MATLAB program, the link for which is given over here okay.

(Refer Slide Time: 01:33)



MATLAB Files: Scripts vs. Functions

- Scripts
  Files containing sequence of MATLAB commands
- MATLAB statements are executed as if typed on command prompt

- Functions
  Files that take certain input(s), executes sequence of steps, and returns output(s) at the end
- MATLAB statements are executed in function's own variable space

So as I said there are 2 types of MATLAB files, script files and function files. So what we have done in the previous 2 lectures is, we have written small snippets of codes in order to calculate multiple things. For example Fibonacci series or the height of the ball at various times, so on and so forth okay. All that we have done in those 2 lectures, we have done it using what is known as MATLAB s script files.

So what are the script files? So script files are nothing but files that contain sequence of MATLAB commands and these sequence, these commands sorry, are executed one after other as they were typed on the command prompt itself. So let us look at the files that we had created.

(Video Starts: 02:21) So this was the file that we had created for the trajectory of a ball thrown vertically upwards. So these commands are executed one after the other. When we click on run or we type the name of the file and press enter in the command prompt. Likewise we can look at fibo using while or fibo using for. These are what are known as script files.

The difference between script file and a function file is that function file starts with the first line, starts with a command called function. So that the syntax for a function file is going to be the function space followed by [ ] separated output variables. So say out1 out2 and so on equal to the name of the function myFunctionName or any name that you choose followed by input variables in 1, in 2 so on and so forth okay.

Keep in mind that the input variables are in circular brackets, the output variables are in square brackets. Having an output variable or an input variable is optional. So if you do not have output variable the function is just going to look like function myFunctionName with input variables which are comma separated. Input variables are also optional. If you have output variables but no input variables, you can just say out1 out2 equal to myFunctionName and let us say you do not have either input variables or output variables. You can say function = myFunctionName okay.

The way MATLAB functions and script work are something that I am going to cover in this particular lecture. But at any point of time if you feel the need to write a function of this type, you should really not be using a function but what you really instead need is script okay. When you save this function okay, MATLAB will ask you to save it with the exact same name as the name of the function.

So keep this in mind as far as this particular course is concerned the name of your function file will be the same as the name of the function that you are using in that function file okay. This is going to work for all the functions that we are going to use in this particular lecture okay. I am going to close this particular function file. Go back to MATLAB and delete that file name myfunction file. I will just click right click over here and click on delete and that will delete that particular file okay. (Video Ends: 05:17)

So function files are file that taken certain inputs, execute a sequence of steps and return the output at the end okay. The MATLAB statements are executed in functions own variable space okay. And first we exit that function all the variables used are no longer accessible in MATLAB.

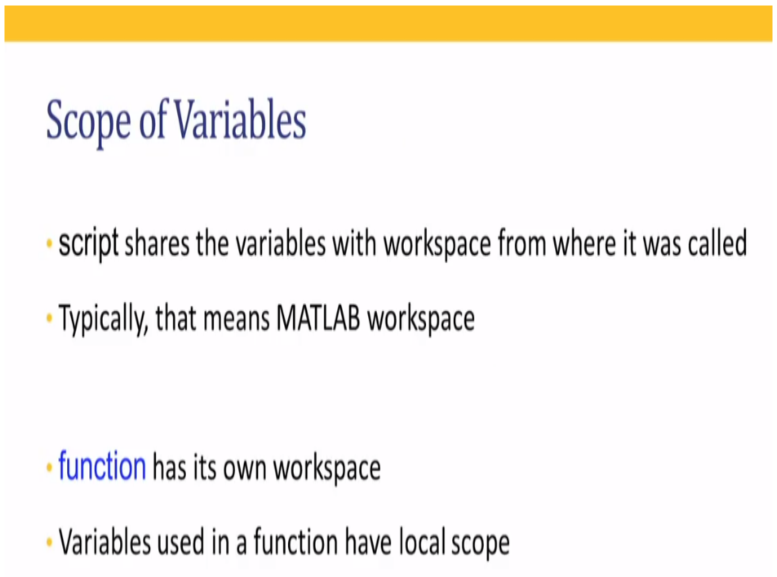
(Video Starts: 05:40) Let us go back to MATLAB and take the example of Fibonacci using for loop okay. And let us execute that particular command. I will clc and I will clear this screen and type fibo using for. And where I press enter, let us look at all the variables that were there. We had declared variable n, declared variable fibo and variable i. when I will run that particular comma run the particular file all the 3 variables that were used in the script file are also available in the MATLAB workspace as can be seen over here.

The reason is because a script file shares the same workspace as the thing that was used to call that script file. Because we call that script file from the command prompt the variables used in the script file are also available in the workspace okay. So script file is nothing but the commands as if they were executed one aft the other in the command prompt okay. It will do the exact same thing.

Let me clear this and what I will do this, do now is convert this into a function file. In order to do that all I need to do is, say function fibo using for and save this okay. I am just doing this to show you how a function differs from a script, this is something you should not be doing, you should not be using a function in this particular way because we want in this particular example we want to execute the commands one after the other and for this purpose we need to be using scripts okay.

So let us go to MATLAB and now type fibo using for again okay. When we type this and press enter you see none of the variables are accessible anymore. If I type fibo that variable is not accessible because that variable has a scope and a scope of that was in that particular function. So when this function file is called n is defined, fibo is defined, i is defined computations are done when we exit that function n, fibo and i are completely gone and no longer available in the workspace. If we want to access that in the workspace, we need to use input and output arguments with the function okay. (Video Ends: 08:13)

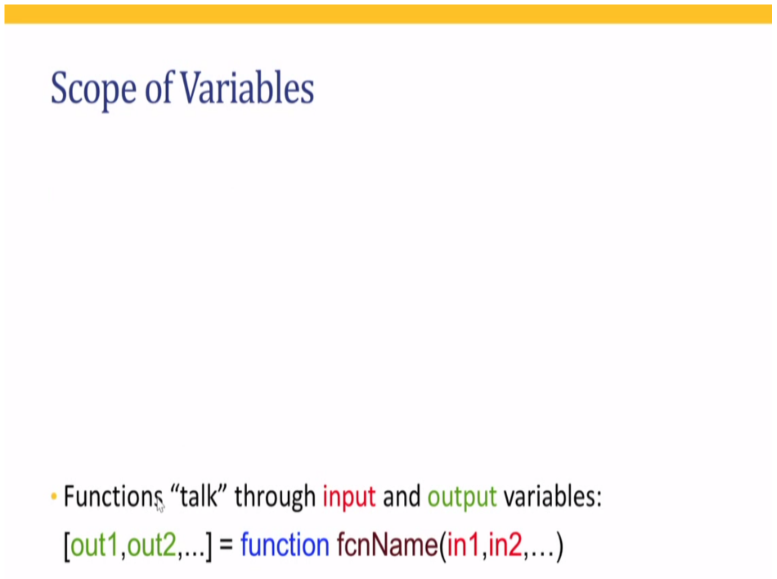
(Refer Slide Time: 08:14)


Scope of Variables

- script shares the variables with workspace from where it was called
- Typically, that means MATLAB workspace
- function has its own workspace
- Variables used in a function have local scope

so that comes brings me to the next aspect scoping of the variables. It is a scripts shares the variables with the workspace okay. Whereas function uses the variables in its own workspace. So the scoping of the variables used in the function they have the local scope the scoping of the variables in the script is the scope is the same as the function on the workspace calling that particular script okay.

(Refer Slide Time: 08:46)

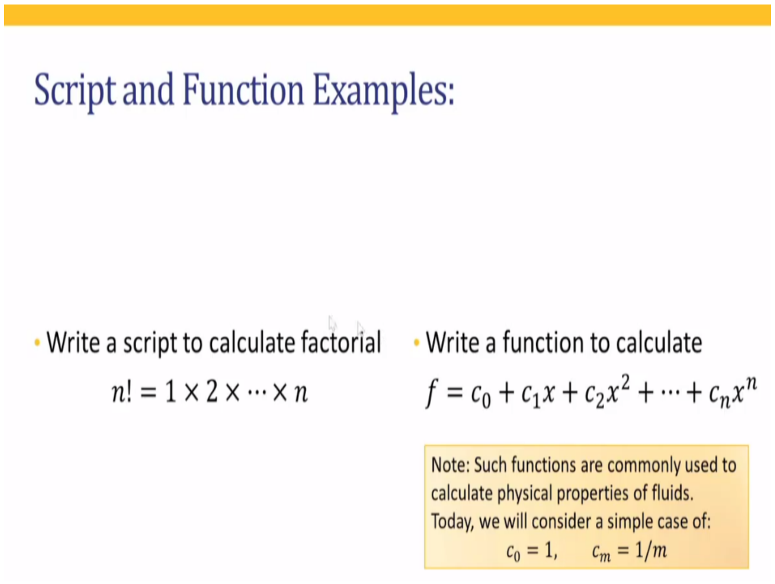

Scope of Variables

- Functions "talk" through input and output variables:

```
[out1,out2,...] = function fcnName(in1,in2,...)
```

And this functions, I will going to talk to the workspace, to the input and the output variable, so as I shown earlier the syntax for a function is basically going to be output equal to function name and inputs that there is an error over here. The function should come before this and not after the equal to sign okay.

(Refer Slide Time: 09:16)



Script and Function Examples:

- Write a script to calculate factorial
  $n! = 1 \times 2 \times \cdots \times n$
- Write a function to calculate
  $f = c_0 + c_1 x + c_2 x^2 + \cdots + c_n x^n$

Note: Such functions are commonly used to calculate physical properties of fluids. Today, we will consider a simple case of:
$c_0 = 1, \quad c_m = 1/m$

So, we wanted to calculate. Let us say calculate the factorial because this is a single purpose thing we are going to write a script. In order to calculate the factorial but if you wanted to write a function in order to calculate a function of this sort, we are going to write a function. So let us go ahead and write a script for calculating the factorial.

(Video Starts: 09:33) So let us say edit myFact and that is going to be a script okay calculate factorial of n. So n equal to let us say 6, fact, value equal to 1 that is what we are going to start with. For i equal to 1:6, factValue = factValue multiplied by I, end okay. So this is a script because this does not start with function call. I am going to go to MATLAB and type myFact.

And all the variables will be accessible in the workspace as we are shown before and factValue is going to be the factorial value 70 okay. The other thing that I would want to do is, I want to make this a bit more, the code to be more better. I do not want to use for loop in doing this instead what I am going to do is, use the array function called product. So if I give prod 1:6 that is going to calculate the factorial for me.

So I will go over, what this particular command does in a bit that is, let us execute myFact in MATLAB and see what let us clear the variables. First clear all and clc okay. So as you can see in the workspace they are no longer. Any variables I give the script name my fact and press enter, I now have the 2 variables fact value and n. The fact value is 720 as we had seen before okay. So, let us see what the command product does. Prod, let us say 1 to 6 is what is that going to do is, calculate the product of this particular vector.

The vector 1 to 6 is going to be nothing but 1, 2 3, 4, 5, 6 so just type that particular thing here. That is nothing but 1, 2, 3, 4, 5, 6 prod which is operated on. This is going to do 1 multiplied by 2 multiplied by 3 multiplied by 4 and so on up to 6. So that is what exactly I am going to get as 720 and that is what this script myFact is doing over here okay. So that is the script that was the purpose of this script is to run or execute certain commands in MATLAB command prompt okay. Now instead if I want to calculate a given function and (Video Ends: 12:32)

The function that I wanted to calculate was c0 +c1x + c2x square up to cnx to the power n. For that I am going to u use a function file (Video Starts: 12:39) I will call this edit myFunc okay and function result = myFunc n okay and what I wanted to do, let us go back over here. (Video Ends: 13:01)

I wanted to calculate c0 + c1x + c2x square and so on. For c0 = 1 and c1 = 1 / 1, c2 = 1 /2, c3 = 1 /3 so and so forth okay. (Video Starts: 13:20) So what I will do is, I will create a vector of the coefficients. Let us call this as a vector C = 1 comma I will put this in square brackets, 1 divided by and I have element by element division 1 to n okay. So let us go over this once again okay. So c, I will just say equal to c not which is 1.

And after that I need to append 1 /1, 1 /2, 1/3 and so on. So let us define a vector = 1 to n. So our c is just going to be equal to c, 1. /vec .You recall in the second lecture of this module, we had introduced the element by element operation ./.

So, what this is going to do, this is going to be do 1 divided by the first element of the vec, 1 divided by second element of the vec, 1 divided by third element of vec so on and so forth. That is going to create a vector, so we are going to have a vector 1 followed by 1 / 2 followed by 1 /3 up to 1/ n. So that gives a vector of coefficients. (Video Ends: 14:53)

So now I have c0, I have c1, I have c2 and so on ,what are the multiplicands over here is going to be x to the power 0, x to the power 1, x to the power 2 and so on up to x to the power n.

(Video Starts: 15:11) So forth that let us define a vector a. So a is just going to be equal to x to the power and for that power, I am going use dot caret vec okay. So that is what I am going to get over here. What I need is the first guy is nothing but 1, x ^ 0 is nothing but 1. So, I will append that over here so 1, x to the power vec okay. And my result is going to be result = C. * a okay.

So the vector C multiplied by the vector a, so that is going to be my result and I will end this with end. The command end is an optional in a function okay. And I am going to save this as the file myFunc okay. You can see over here that this myFunc is actually highlighted. The reason is because I have made an error in naming this the file name is my capital func whereas name of the function is small func okay.

Recall that in MATLAB, all the commands are a case sensitive. So I want to have the function name same as my file name. So if click on fix it will change the function name to capital f. I will just save this and go to MATLAB and run myFunc, myFunc n and let us say result = myFunc n where n was equal to 6 okay. Okay I get error over here that is because of not defined my x. So, I need myFunc n, x over here.

And I need to call it with 2 arguments n and x. So n was 6 and s let us say x was 0.1 and I call this particular function and I am going to get the result equal to sum result and that is going to be sum of so it is going to be (1+ 0.1) / 1, 0.1 square divided by 2, 0.1 cube divided by 3 so on and so forth. And that is going to be my result okay. So see what are what does happened is while doing this particular function I have made an error that error was, (Video Ends: 17:58)

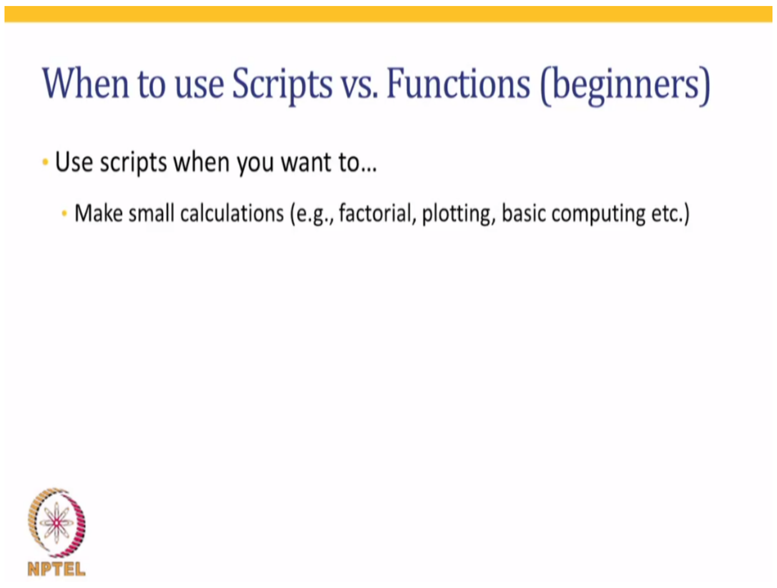
I am calculating c 0, I am calculating c1x, I am calculating c2x square and so on up to cnx to the power of n. But I have not summed at up in order to calculate the function f, okay. (Video Starts 18:11) I will go in over here and just change this to say sum C *. a .Save this and now I will call this clear clc okay. And call result = myFunc 6, 0.1 and I am going to get the result over here okay.

Inadvertently what I ended up the show, ended up showing over here also is the way we will do debugging. We recall that we got 2 errors, 1 was an actual error that stem because I had not defined my input variable x at all which basically meant when it came across this particular command what happened was, it was not able to calculate because x was not defined.

Let me go back to how I had return earlier and run this again okay. I will run this again and I got that error undefined function or variable x on line number 6. I went to that function on this particular line and realize that I had not defined my x, so I went ahead and made that change defined x and I call that particular function using 2 arguments as shown over here.

The other error that I had done was, I was calculating this particular array C * a, but not summing that up. When I saw that result which resulted not in a single value which was the sum of that elements of that array but which was just the 6 elements of that array. I realized that I needed to add that sum command over here okay. So this is also how we do debugging in MATLAB. (Video Ends: 20:17)

Ok so let us recap. When to use functions and when to use scripts and for beginners that is for most of you guys these are the general guidelines that i have for you okay.
(Refer Slide Time: 20:33)

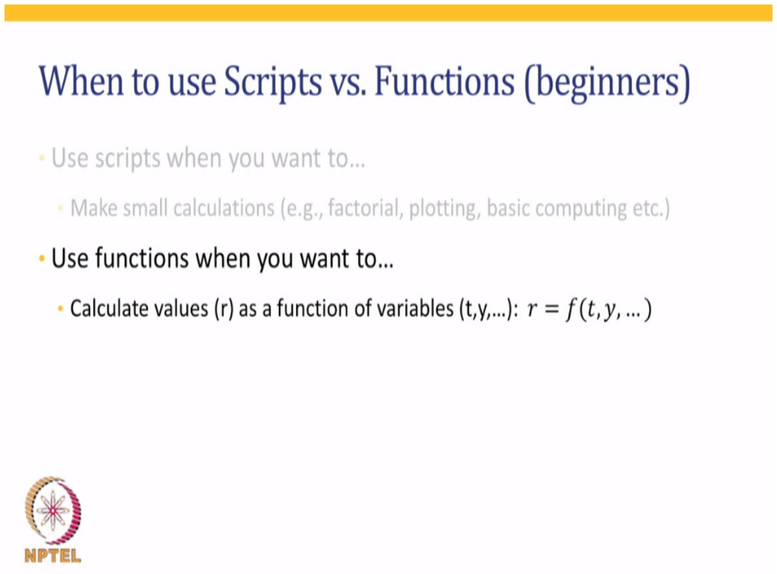


When to use Scripts vs. Functions (beginners)

- Use scripts when you want to...
  - Make small calculations (e.g., factorial, plotting, basic computing etc.)

So you want to use scripts when you wanted to make small calculations such as factorial or basic computing as we did in the markList example in the previous lecture or when we wanted to do plotting of some certain functions which we are going to cover in the next lecture that is where we want to use scripts.

(Refer Slide Time: 20:53)



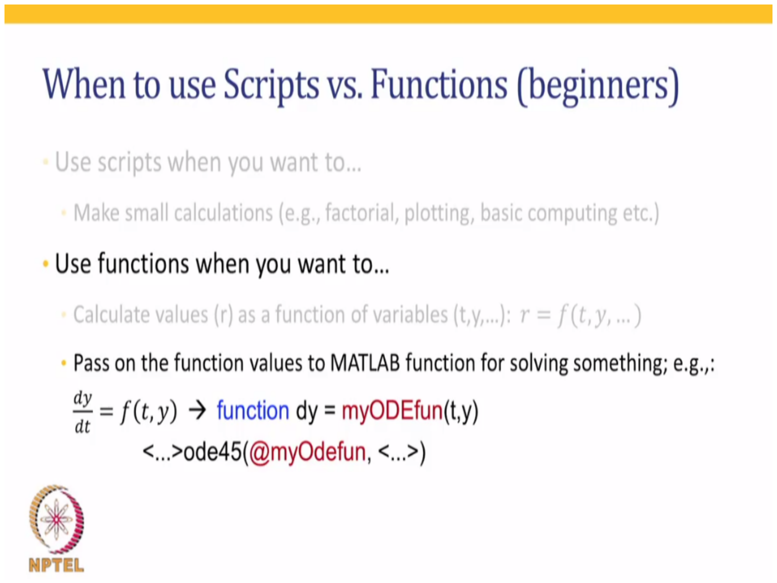
When to use Scripts vs. Functions (beginners)

- Use scripts when you want to...
  - Make small calculations (e.g., factorial, plotting, basic computing etc.)
- Use functions when you want to...
  - Calculate values (r) as a function of variables (t,y,...): $r = f(t, y, \dots)$

NPTEL

When do you want to use functions, let us say we want to calculate the values r, a result as a function of variables t, y, so on and so forth when we want calculate r as a function of t , y and so on. That is where we want to use functions.

(Refer Slide Time: 21:09)



When to use Scripts vs. Functions (beginners)

- Use scripts when you want to...
  - Make small calculations (e.g., factorial, plotting, basic computing etc.)
- Use functions when you want to...
  - Calculate values (r) as a function of variables (t,y,...): $r = f(t, y, \dots)$
  - Pass on the function values to MATLAB function for solving something; e.g.,:

$\frac{dy}{dt} = f(t, y)$ → function dy = myODEfun(t,y)

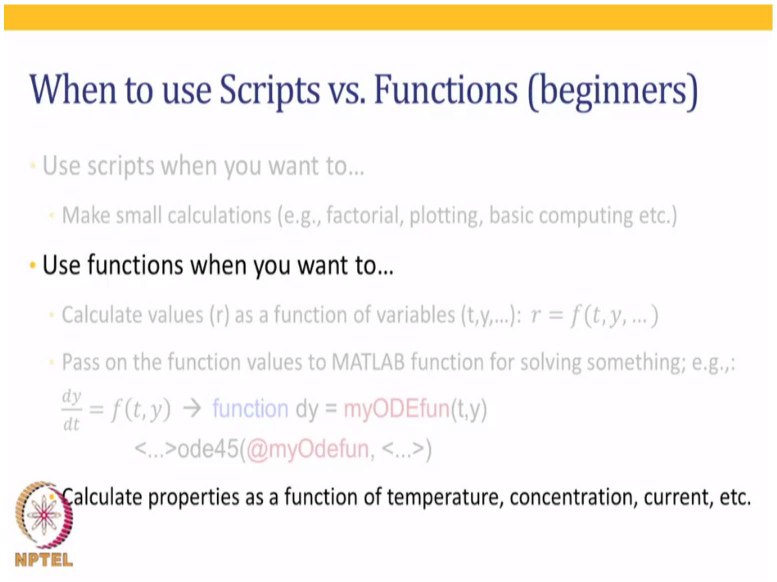
```
<...>ode45(@myOdefun, <...>)
```

NPTEL

The second example when we want to use functions is, when we want to pass them as a to MATLAB functions. For example, in ode solving example that we saw in Dhoni hitting the 6. We are calculating function using ode 45 in MATLAB and we wanted to pass that f of t, y as a function to ode 45 that is another example where we are going to use functions.

This is something that we are going to cover in module 3 and thereafter we will not even cover this in module 2. Just giving you a brief overview of when to use functions, third example, when to use function is, let us say we want to calculate something as a function of temperature.

(Refer Slide Time: 21:44)



When to use Scripts vs. Functions (beginners)

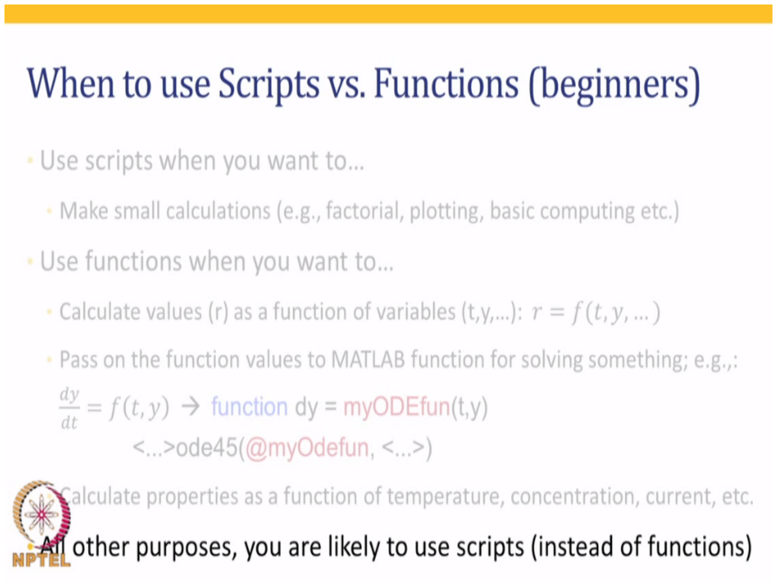
- Use scripts when you want to...
  - Make small calculations (e.g., factorial, plotting, basic computing etc.)
- Use functions when you want to...
  - Calculate values (r) as a function of variables (t,y,...): $r = f(t, y, \dots)$
  - Pass on the function values to MATLAB function for solving something; e.g.,:

    $\frac{dy}{dt} = f(t, y) \rightarrow$ function dy = myODEfun(t,y)

    <...>ode45(@myOdefun, <...>)
  - Calculate properties as a function of temperature, concentration, current, etc.

Let us say we want to calculate for example the saturation pressure of steam at various temperatures or we want to calculate the a let us say diffusivity of something as a function of concentration or let us say we want to calculate something as a function of voltage and current, so those properties which we want to calculate as a function of let us say temperature or concentration current or current so on and so forth.

(Refer Slide Time: 22:20)



When to use Scripts vs. Functions (beginners)

- Use scripts when you want to…
  - Make small calculations (e.g., factorial, plotting, basic computing etc.)
- Use functions when you want to…
  - Calculate values (r) as a function of variables (t,y,…): $r = f(t, y, \ldots)$
  - Pass on the function values to MATLAB function for solving something; e.g.,:

    $\frac{dy}{dt} = f(t, y) \rightarrow$ function dy = myODEfun(t,y)

    <…>ode45(@myOdefun, <…>)
  - Calculate properties as a function of temperature, concentration, current, etc.
- All other purposes, you are likely to use scripts (instead of functions)

That is when we will use functions so these are the 3 cases when we use functions for all other examples for all other purposes we are more likely to use scripts rather than function okay. So, with that I come to the end of that of this lecture what we covered in this lecture is the examples of when to use scripts and when to use functions in order to computations.

Scripts is nothing but sequential execution of certain commands as if you wanted to the execute them in the calling function or in the calling workspace, functions are to be used for achieving a specific goal when you want to calculate for example r as a function of t, y and so on and so forth okay.

So these are main uses of functions and script in MATLAB we are going to rely heavily on using functions and scripts in this particular set of lectures from module 2 onwards okay. So with that I come to the end of this particular lecture, the next lecture is going to be the last lecture in the introduction module the next lecture is going to covered plotting and program output thanks and see you in the next lecture.