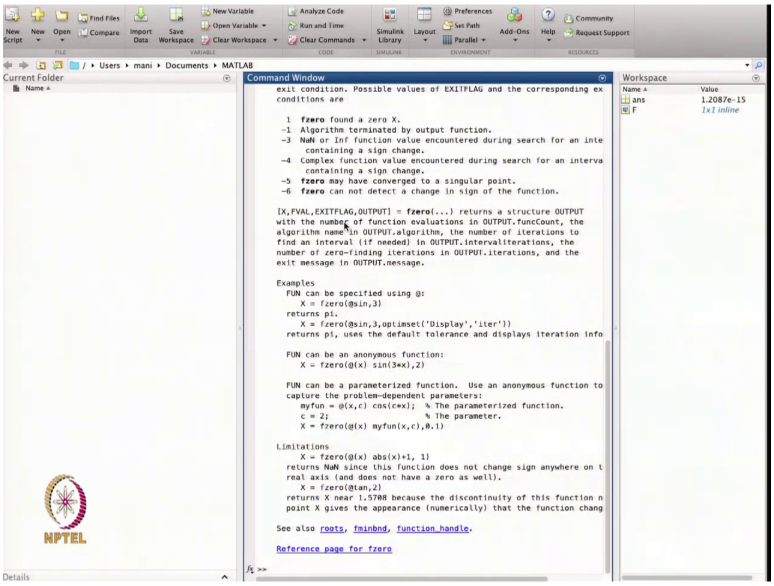
Scientific Computing Using Matlab
Prof. Vivek Aggarwal and Prof. Mani Mehra
Department of Mathematics
Indian Institute of Technology, Delhi
Lecture No. 05
In continuation of basics of Matlab

Welcome back to this course, today we are going to do the lecture 5 and the basics of the Matlab. So, let us start with the Matlab. So, this is the Matlab.
(Refer Slide Time: 0:33)

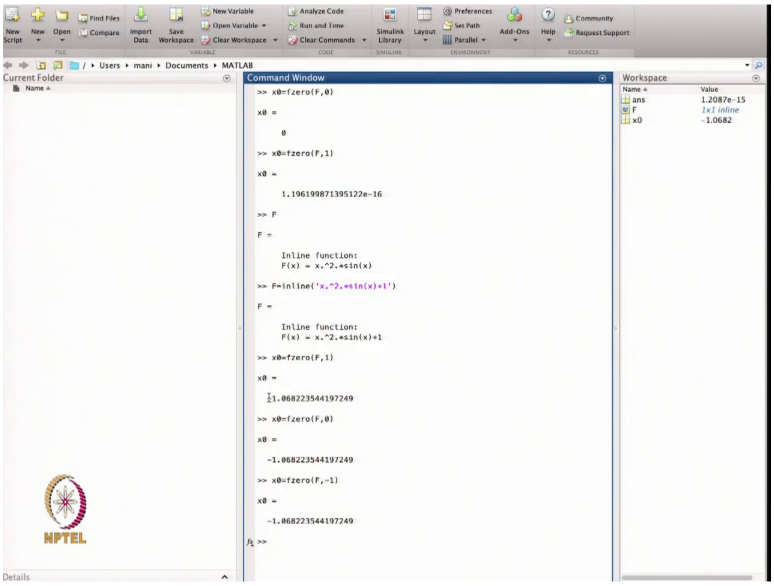


So, in the previous class we have started with how we can define the functions anonymous function and inline function. So, in the last class, we have just started the function. So, today I will give you the definition of another command. So, we are going to do today how to find the root of a function of a maybe equation like I have a function.

So, in the last class suppose I I have defined the function f that is equal to inline function I have just defined, maybe x^2 sin x. So, this is the function I have defined in the last class the x^2 sin x and then we have discovered that I can define this find the value of this function at π or with some other value.

Now, suppose I take a command that is called fzero. So, this is the command fzero. So, what did the fzero, fzero is single variable nonlinear zero finding. So, it will give you that suppose I have a function and you want to find the zeros of that function maybe the roots of that function.

So, this command gives you so, that is the notation of this command that fzero and whatever the function we are defining and x_0 is it gives you it will find out the root near close to this value x_0.
(Refer Slide Time: 2:33)

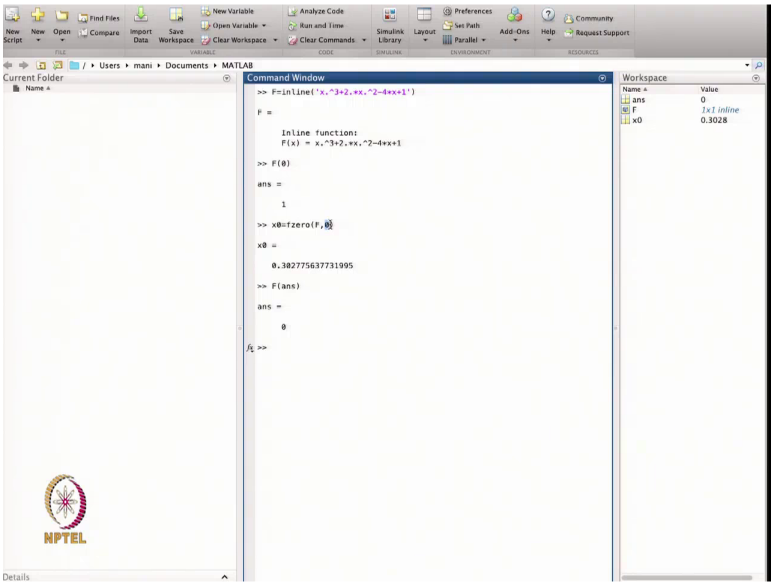


So, suppose the same function I want to do and I want to find out what is my root, so I just write it x_0 that is equal to fzero and my function f, just now I have defined and I want to find a 0 value of this function near 0.

So, it gives you the value of 0 or maybe I can define this one. So, that is again the value 0. So, it means the function capital F, we have defined. So, this function has the value 0 as the root of this function, maybe I can define another function. So, this is what I defined another function in this or I will change to plus 1. So, that is another function I have defined now I want to find its zeros.

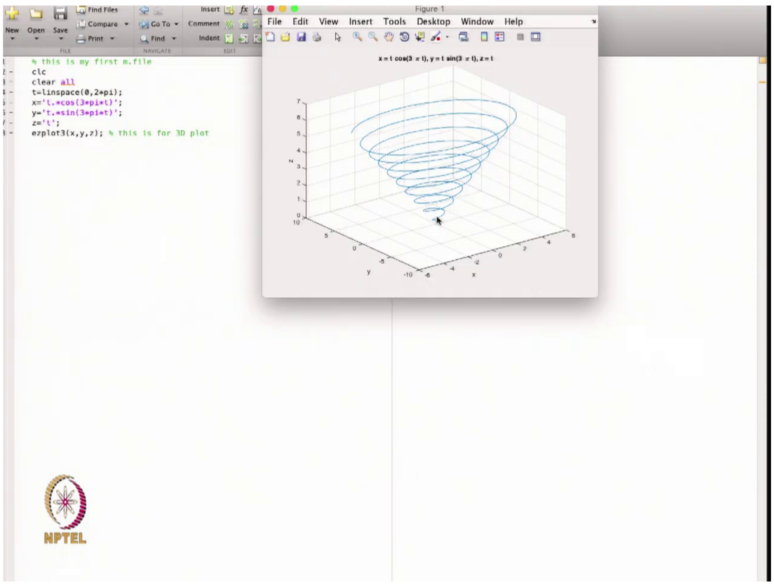
So, this is a 0 I have defined. So, now I am defining the zeros finding the zeros of this function near x = 1 and it gives you the value -1.068. Similarly, I can define the value near 0, so that is another same value I am getting, it means that the roots of this function are close to 0 or 1 the same as minus this one or maybe I can define -1. So, it keeps the -1. I can define a polynomial like so I define just another function.

(Refer Slide Time: 3:59)



I just define the function, another function capital F. And now in this case, I define a polynomial q + 2 * 2 dot star into x^2 -4x+1. So, this is a function, I am just defined, this is the polynomial that we have defined x^3+2x^2-4x+1. Now, I want to find the value of this function at maybe 0 just check. So, the value is coming 1. Now, I want to find the zeros of this function a polynomial.

So in this case, suppose I want to find the value of the polynomial near 0. So, that is the x = 0.3. So, if I define the value of the function at 0.3 or maybe answer. So, that is the value it means that x_0 is the root of this function. So, by the help of this fzero command, we can just find the root of the given function. What are the functions we have near the point because here we have to give the initial approximation that in which vicinity we want to find the root. So, this way we can find the root.
(Refer Slide Time: 5:29)



So, now to go further, this is all about the function. Now, we can define how to define the script file. So, let us start with this one. So, in this case I will go to this new and here it will give you that you what do you want to find out so, I want to write a script file here. So, this is the script file. So in the script file, whatever the command, I have typed on the command windows, the same thing I can do in the script and I can run the whole commands together after writing in this script file, so let us start with this script file.

So, this is my first m file. So in this case, and I always start with clc and clear all so clc means whatever the written on the screen when I run this code, it will clear all the commands on the command window, and it will clear all the variables whatever is available in the workspace.

Now maybe I can define if a function variable t, so, I just define t=linspace(0,2π) with 100 number of default points, so that in my value of t, now, I would define the other function the x(t). So, x(t) I just I can define by the way that So, I just define x = t cos(3πt). So this is a function I am just defining. So, that is my x.

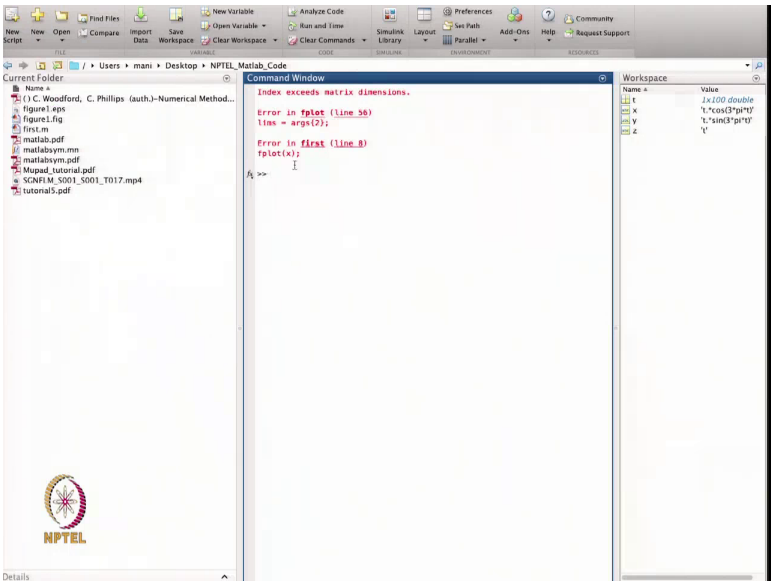
Now, I define the y as equal to maybe the same way. So, t star sign this is a vector, I have to define sin(3 πt). So that is my y. And then I define the z=t. And that is it. And now, suppose I want to plot this one, so I will write ezplot. So, ezplot so in this case, I have 3 variables. So, do the 3D plot.

So, I will write ez 3 and then I will define xyz this one. So, this is I just in the code I can always write the commands, which is just for my understanding that this is for 3D plot to 3D plot this, so, this is the function we have defined the parametric function in in the 3D that is a function of t xyz and then I want to plot this one.

So, now suppose I want to run this one, but before this one I have to save this function. So, it is save as. And I can define here, the first dot m. So, this is a command you can change the directory also maybe I can define here and nptel code and then saving this one now I can run this code. So, it will ask you to change the folder. And now it gives you the answer.

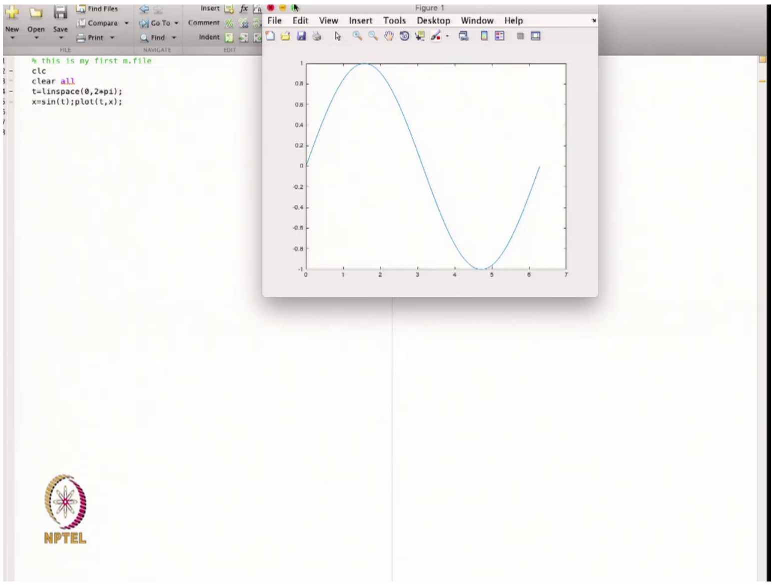
So that is the ezplot we have just defined. And that is the spiral, you know that this shows you a spiral. So that gives you the plot of the spiral function we have just defined here. And this is the value the function. So x =tcos(3πt), y=tsin(3πt), and z = t.

So, instead of writing each command, again, separately, and then finding the final command that is a plot, but instead of that one, the all commands we can find in the script file, and then we can run the script file together and then we can have this one in the single code. So, that is the script file. Similarly, I can define another function. So, I can change this function instead of this. Maybe I can.
(Refer Slide Time: 11:03)



Now I can define here I can maybe define fplot. So, fplot is what it gives you the plot of the one function only. So, in this case maybe I can define fplot of x just x. So, that gives you that plot. So, fplot gives you that this is the error.

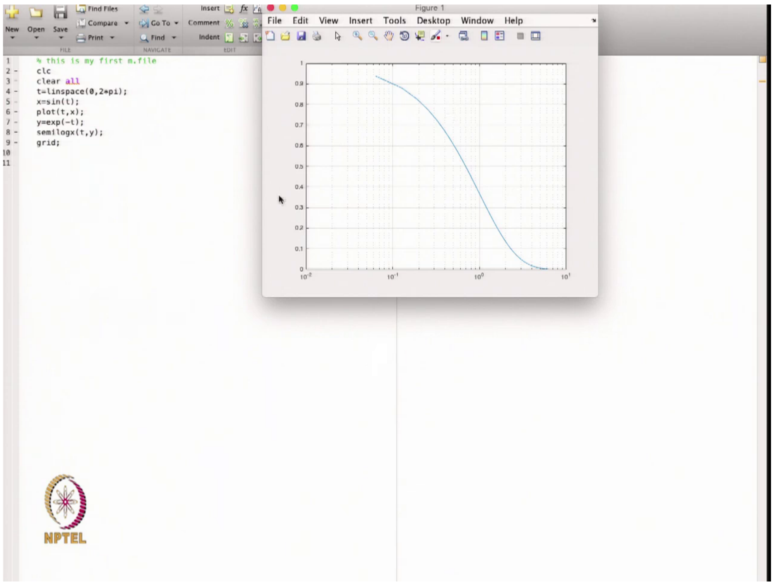
(Refer Slide Time: 11:34)



So, I can define so fplot is this one and this is a t function I have defined now I can function x = sin t. So, these are functions I have defined. Now, I can plot this function this is t and this is maybe x and in this case, so, now I can plot and I can run this code. So, that is a plot I got.

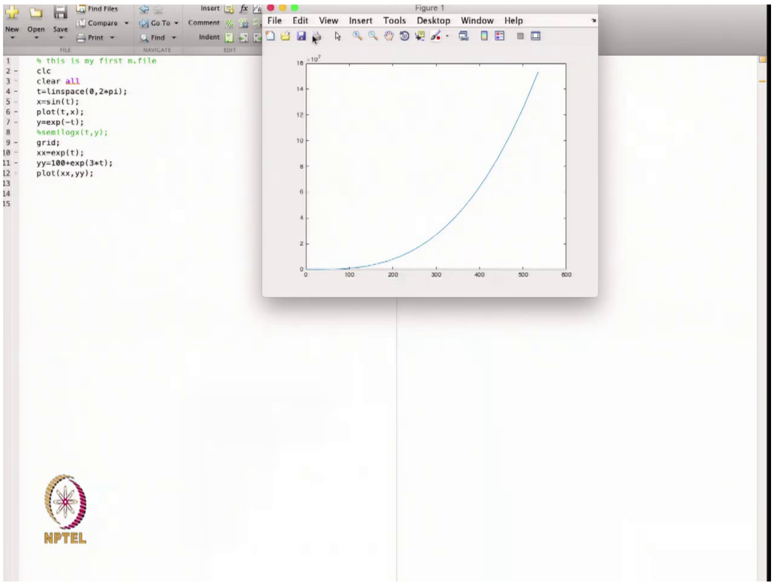
So, instead of writing that function in the inline form or the function form, I can define this as a vector form. So, this x = sin t and I can plot this function or this one.

(Refer Slide Time: 12:26)



Maybe I can define the another thing I have I define another function maybe y is equal to exponential is power minus t. So, this is a function I just defining and t=0 to 2 π. Now, what I do is that I use semilog so semilog is then I will write the semilog(x). So, it gives you t x or t y. And then I can define a grid. So, that is a function we have just defined, so semilog(x) means it gives you the log at that x axis and value the function and the y axis. So, this is the log x.

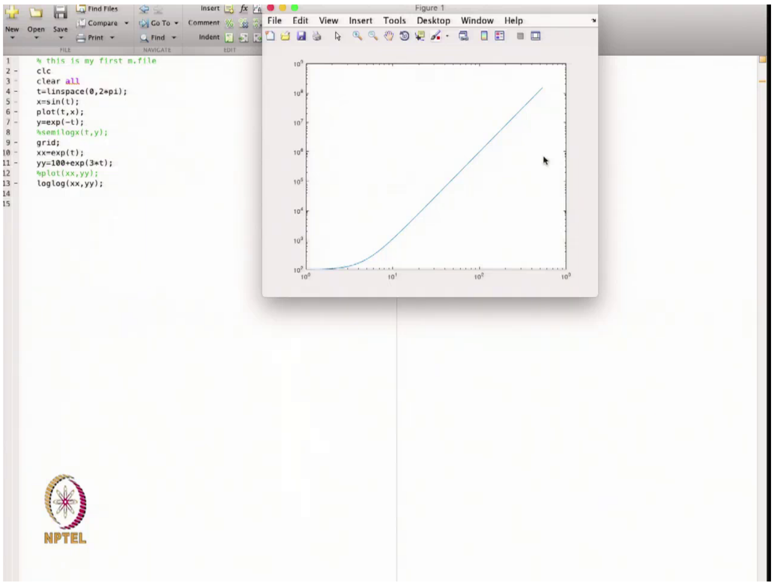
So, it is a 10^{1} ,10^{0} ,10^{-1} , 10^{-2} . So, it gives you the x axis as a log value and the y axis as just the linear value, the real numbers . Maybe I can define another function from here. So, I define just x is equal to this.
(Refer Slide Time: 13:57)



So, let us define another function xx, so, the xx $=e^t$ and then I define another function yy=100+e^t. Now, I want to plot this function.

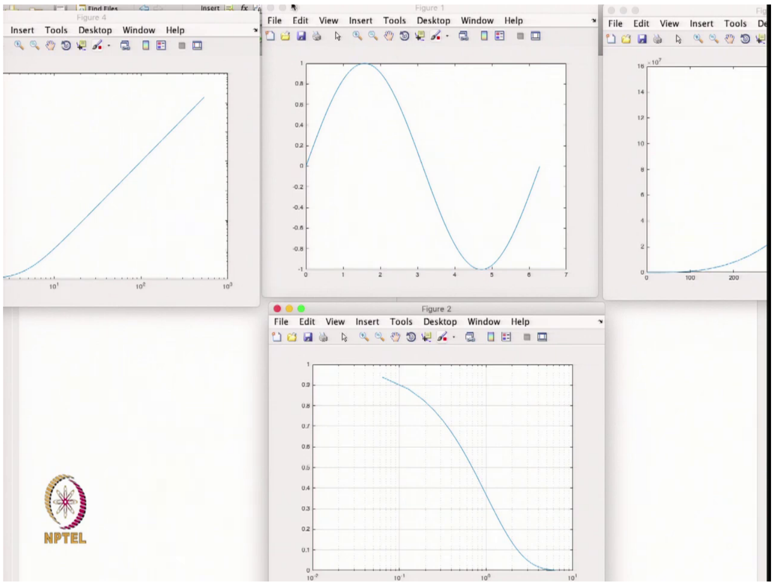
So, if I define the plot xx and yy this one I just put the sign here. Now, this is the plot we have got. So, from here you can see that x-axis the value of the X is 100, 200, 300 up to 500 and the value of y-axis also very large, it is going to give you the value $2*10^{-7}$.

So, basically if we want to if we are dealing with the large values on the x axis and on the y axis. So, instead of plotting the function I can write down here loglog plot. So, this is a loglog plot.
(Refer Slide Time: 15:23)



So, now, I can define xx yy and here I can just command put the. So, that is the same function we are plotting here, but now it is given to the loglog. So, it means that the x axis is giving you the log value of that and y axis also the log value of that. So, in this way it is very easy to understand that this is a function. So, whenever you are dealing with a large value, it is always better to plot in the loglog form.

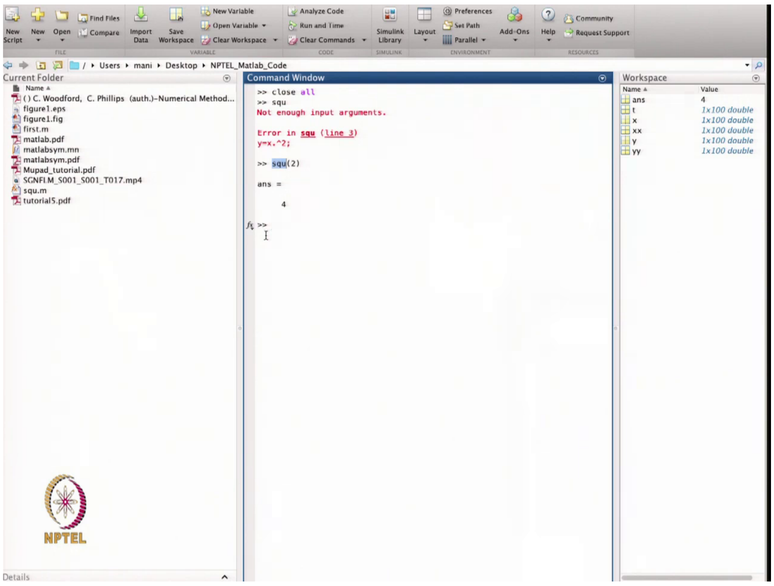
So this all these things you can do in the same file or maybe suppose I want to have all these plots together.
(Refer Slide Time: 16:11)

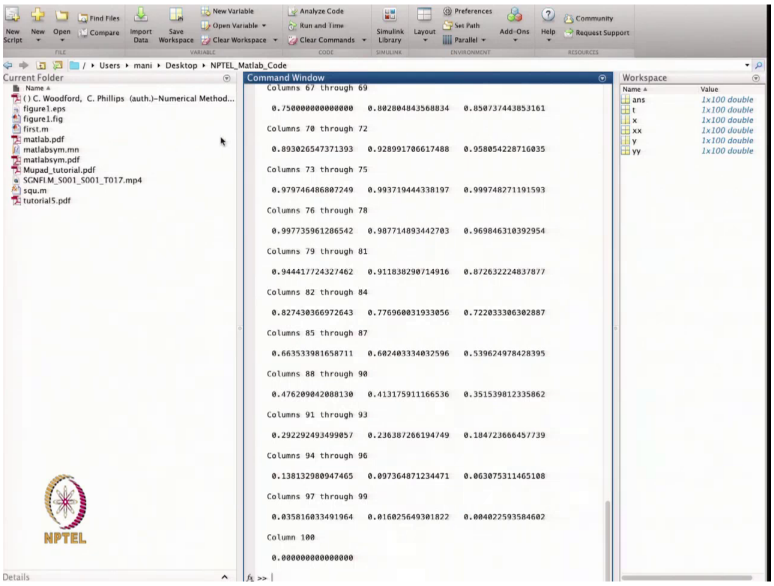


So, in this case what I will do I will write figure 1. So, this will be the figure(1) and then I can define figure 2 here and this is another plot. I am just defining figure 3 and this is another plot is figure 4. Now, I can run this one. So, you can see that it gives you all the plots together. So, that is plot 4, plot 3, plot 2 and plot 1.

So, all the plot comes together because I have defined the function (fig) figure each figure for each plot. So, if we want to close this one all together, so I will write just here I will write close all so, it will close all the graphs whatever the plots we have made it by this way so, this script file is basically now you understand that this script file is used whenever we want to compile large number of commands together so, that that is a script file. Now, the same way I can define another file.

(Refer Slide Time: 17:46)





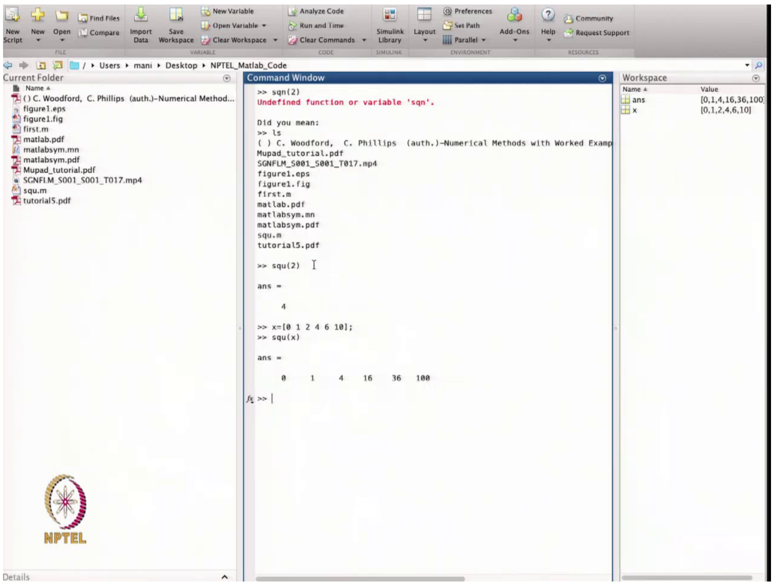
So, let us define another file new and that is the function file. So, that is the function file I can define. So, this is I will start with the function and this is the output I'm going to define and this is the input we are going to define and that is the name of the function.

So I will just write here, maybe. Suppose I define just the value, y. And here I am defining the input, I am giving maybe x. And this function I am defining is sq. Square function. So, just I am defining what I am doing, if you pass the value of x to me, then this function will give you the square of this value.

So, I just write full name or maybe the small name square and that is the, the body of the function. So, here I can define my function y = x^2 , then it will return value, the only condition is that if I want to save this function square, so, I have to first save this function. So ,save as. So, the name of the file.m and the name of the function should be same. So, in this case the automated coming squ.m and I will save this one.

Now, suppose I want to use this function later on. Now, suppose I want to run this code, so, what will happen? So, it gives you the square not enough input argument because you know that in that case, we have to pass the value so, I if I write squ and is equal 2, let us see, so t will give the value answer 4.

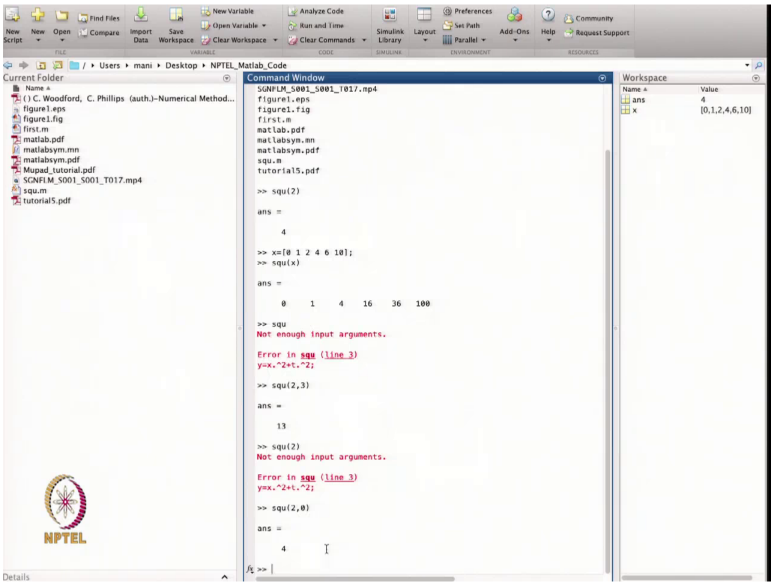
So, now in this case, I am calling the function square from here, giving the value 2 here and then it is giving the value 4 what will happen if I pass maybe so, in this case my x=100. So, now what I will do I will just pass the x here. Now see, it is giving the square of that one or maybe I can clear all clc.
(Refer Slide Time: 20:18)



Now I can pass squ 2. So, I have to first go to that directory where the function is defined. So, let us see what are the files there. So, this is squ. So, I will write squ, because you have to be in the same directory where the function file is saved. So, this is the function file set and I am there. So, this is my value the function now suppose x, x = 0, 1, 2.

So, this is a value of the function. Now I want to do the square of all this together. So that gives you the value of the function, maybe I can define the same function without two arguments.

(Refer Slide Time: 21:16)

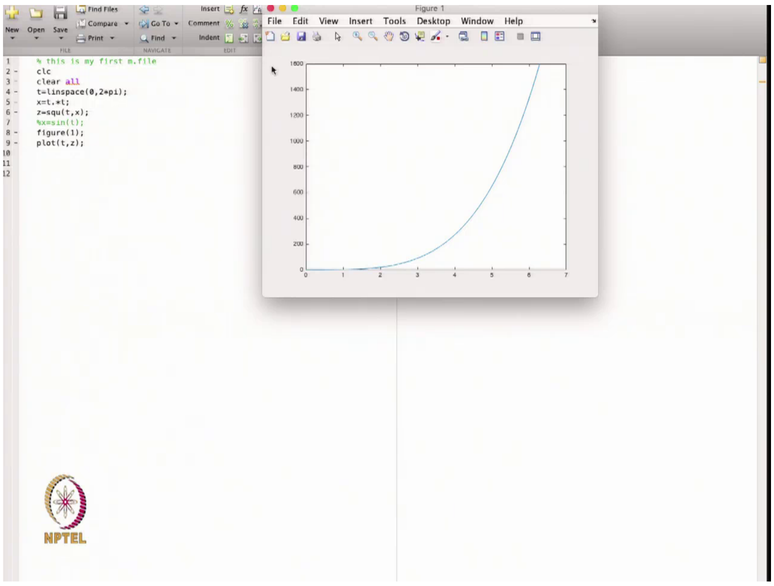


So, let us change this one. And in this case, I just put the two arguments, x and t. So, in this case, I am putting the two arguments. And then I am, here I am defining the squares, sum and the squares. So, I am doing the squares first of each function, each argument and then putting the square and then the sum. So, this is the value I am defining. So again, I have to save. So, this is the same, the name is the same, only the argument has changed.

Now if I want to define this function, this is the function it will be the same. That not enough inputs. Now, I will give you the inputs here. Now, I give 2 and 3, 2 and 3. So, this is the two arguments I have given. So, that gives you the value 30 what will happen if I give them one argument only. So, it is show you that not enough input arguments, then we have to find because it gives you the header or it is give you the body that whatever the function we are calling inside that function this is a defined function.

So, in this from here I can see that I have to give the two arguments. So, I will pass the two argument and it gives you the answer. So, in this way, we can define the function script file and this is the function, we generally call it the function script function file. Or this is the same as we used to do in the sub routines whenever we used to do programming in the Fortran or maybe C.

So, then in that case we have to define the sub routine. So, this is the sub routine basically we're defining. So in this case I can call this function from any script file and that file gives me so let us define clear all clc.
(Refer Slide Time: 23:17)

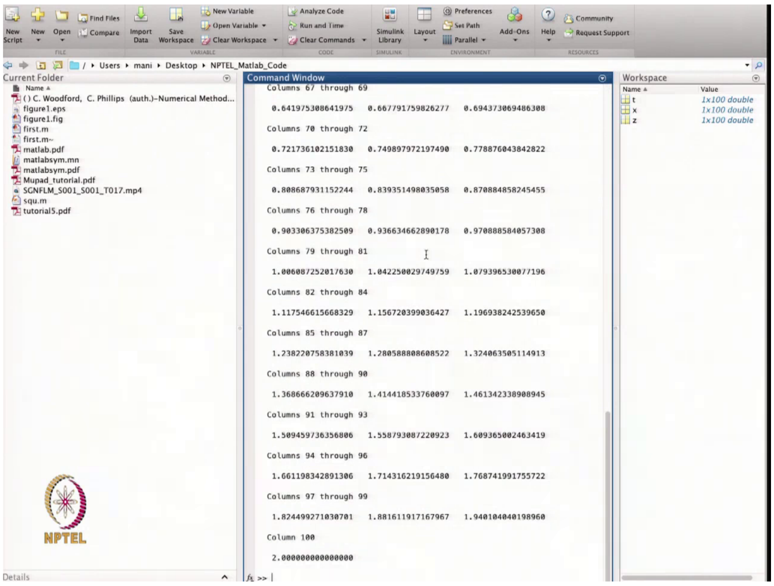


So, I want to see that. So, this file directory contains the function whatever we have script file defined that is first dot m. And this is the function we have defined. So, let us go to the script file. So, where does my script file open. So, open means this is the first file I am defining. So, this is the file. So in this case, suppose I want to use the function. So, what I do this is my t is there and x sin t is there.

So, I just make it maybe t is there and then I will pass this one. So, now I call t is there and I call x is equal to maybe be t lets multiplied by t itself. So, I get this x. Now I do I call the function and I will take another value that is z value I am defining and in this case I will find squ, I just call that function as squ and pass this t and x.

And then I define the figure 1. So in the figure 1, I will plot this t and maybe, z I plot this all other just I get delete. And then I will run this one. So, that is the plot. So see in this case, we have just defined the linspace and t's, I am defining a t from zero to two π maybe I can define from 0 to one in this case and this is the plot. So, I am defining the t from 0 to 1, then I define the x that is t square and then I call the function from here. So, z = 1.

(Refer Slide Time: 25:37)



So, what is the z here? So, it gives you the so, what is the z. So, z is this one So, z is t square plus x square because, if you see the t is this one, because I have a function now, and x is this one, so, let us try this one. So, I just take a t tenth value, square of that plus x tenth value, and then square of that. So, that is a value. And I just want to check that what is that tenth value of z.

So that is the value that this and this, so instead of doing the same things here, I am calling my function from this script file. And that function is used here to find the square of $t^2 + x^2$. And then I am using this function to find the plot in this script file. So the same MATLAB functions can be used in the different different script files, and using these script files.

Using these functions, I can use my codings to do the further coding for whatever the scientific computing I need to do and based on this one, I can always use the same function in various script files. So, that is a all we can do with the function file and the script file. And now, I think we have enough knowledge about the basics of Matlab.

So, now we can start our scientific computing course, and then we can define we can discuss various methods to whatever is given in the syllabus of the scientific computing. And then we can make the Matlab code and used Matlab code we can verify whatever we are doing in the scientific computing. For example, suppose we from the next lecture, we are going to start with that how the errors comes to the computation.

And then we will go on, we will define that how we can define the roots, maybe roots of a nonlinear equation, and in that case, we will discuss the various method like a bisection method or the Newton Raphson, Secant method. So, we will make the code for those methods in the in the Matlab and then we will verify there itself. So, that is all about the basics of Matlab.

So, I hope this is enough for you to start with the scientific computing. So, we will go for the next lecture we are going to start with the scientific computing. So, thanks for viewing. Thanks very much.