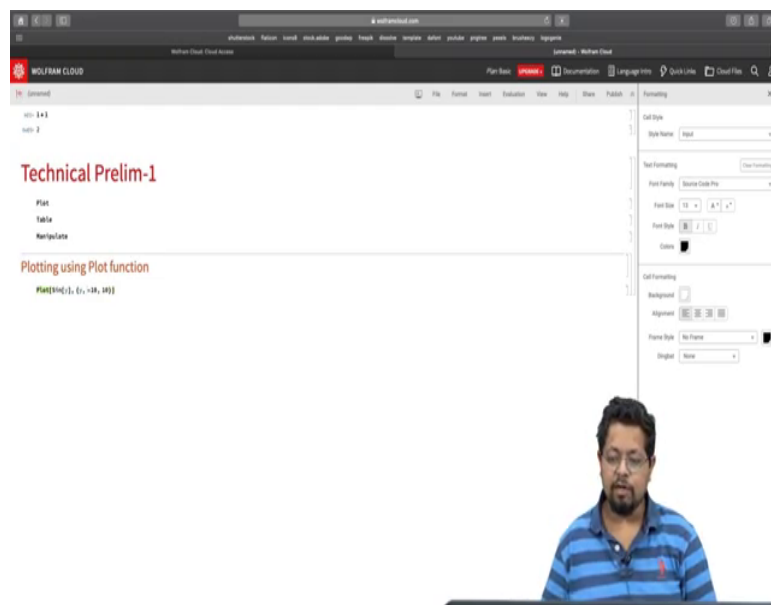


Physics through Computational Thinking
By,Dr.Auditya Sharma and
Dr.Ambar Jain
Department of Physics
Indian Institute of Science Education and Research, Bhopal
Lecture 2
Technical Prelim-1

Hello everyone, last time I gave you a brief introduction to Wolfram cloud. What we will do today is: we will go to Wolfram cloud and start implementing some of the things.

(Refer Slide Time 00:36)



Today is the first technical prelims. So, what we are going to do is: we will learn the basics of few of the functions and see how it is implemented in Mathematica. Once you have opened your Wolfram cloud Mathematica file, you can go ahead and compute something to test that you are online and it is working. Once we are here, today's objective is to discuss three particular functions: one of them is the 'Plot' function, the other one is 'Table' function and the third one is 'Manipulate' function. Today's lecture, we will call as 'Technical Prelims'.

So, we can simply go ahead and introduce a format here, using the 'Format' toolbar. I will introduce a title, we will call this as 'Technical Prelim 1'. So, technical prelims will introduce you to Mathematica functions. It will be basically, understanding how to implement the functions. The technical prelims will not include any discussion about physics or mathematics. They will just be understanding how to implement Mathematica functions.

So, today we are going to talk about 'Plot' function, 'Table' function and 'Manipulate' function, because this is something we will require for the next session. So, let us go ahead and start with the Plot function. So, at this point, I can go ahead and introduce a section.

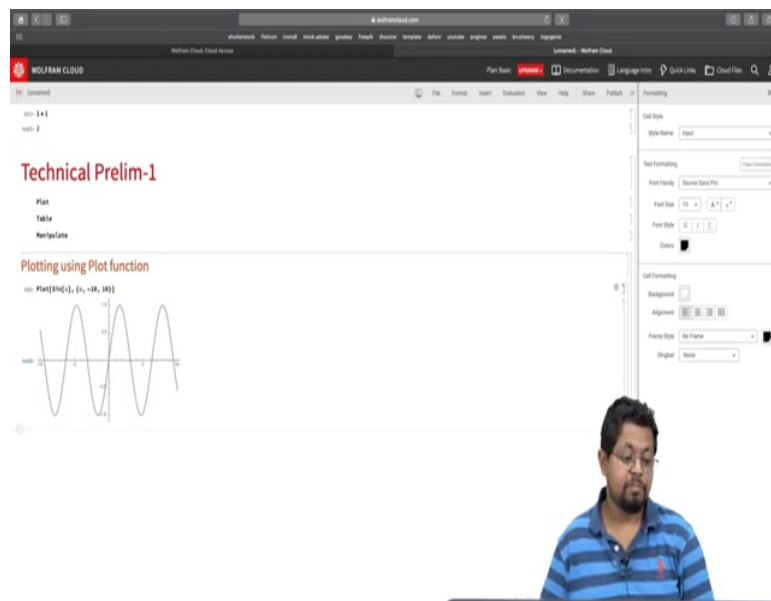
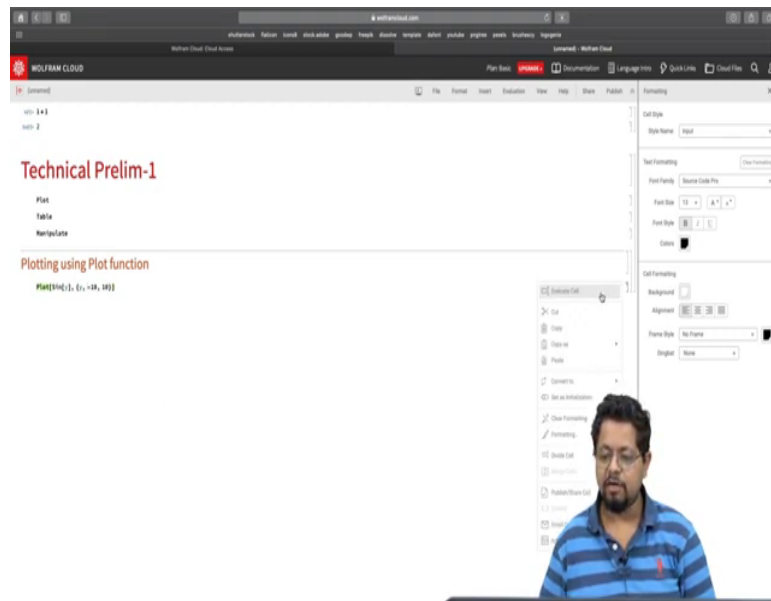
Plotting using Plot function: Plot function is a function in Mathematica, which allows you to plot graphs. So, we will take a simple example. As always in Mathematica, this is the name of the function and the arguments of the function are put in the square brackets. So, I have created a pair of square brackets and inside that I am going to put my function. All the Mathematica functions start with a block letter.

So, in this case, I want to plot 'sine' function and 'sine' function again is a function in Mathematica which starts with the block letter S and the argument of the 'sine' function will be put in these two square brackets. As you hover your mouse over these square brackets, you can see which brackets are closing, in the sine function. I will go ahead and write 'Sin[x]' and then I will say, I will plot this from $x = [-10, 10]$.

Now, note the way I have given the argument over here, this is the second argument of the plot function and 'Sin[x]' is the first argument of the 'Plot' function and the two arguments are separated by a comma. The second argument is specifying, what is the name of the variable in which the plot has to be made and what is the range of that variable?

So, here the variable name is x and this x matches with the x over here and the range: the minimum value is -10 and maximum value is plus 10. We are going really very slow over here so that we can understand some of the basics. Since Mathematica is a symbolic programming language, I can put a 'Sin[y]' over here and call this 'y' and, that does not change anything, it means the same thing. At this point, I will go ahead and evaluate.

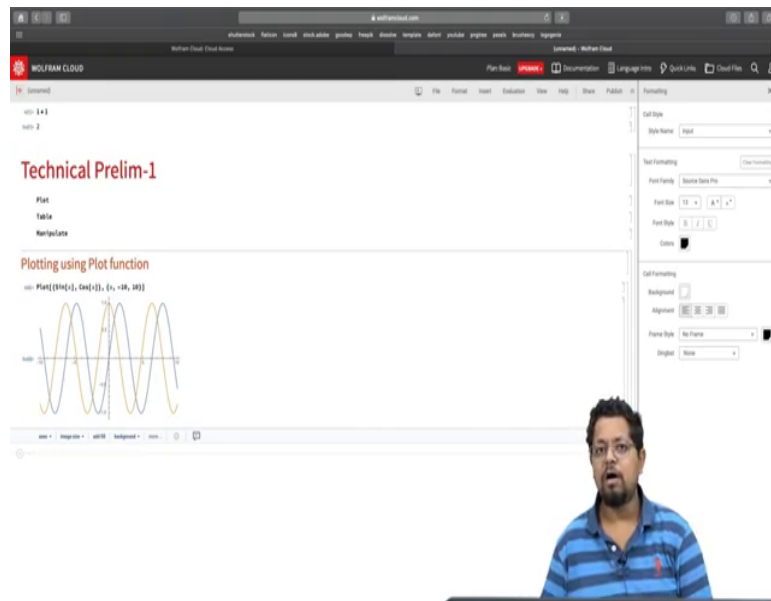
(Refer Slide Time 04:33)



For evaluation, I told you, you can click on this little gear icon over here and you can say evaluate cell or if your keyword allows it, you can press shift plus return or shift plus enter on your keyboard. So, when we plot a 'sine' function, this is what we get and as I said you can call this as anything else. You can call this as 'a', and that is perfectly fine, you will get the same result.

So, this is a plot of the 'sine' function. At this point, I want to compare this function with let us say another function. Let us say, I want to compare this with 'Cos' function.

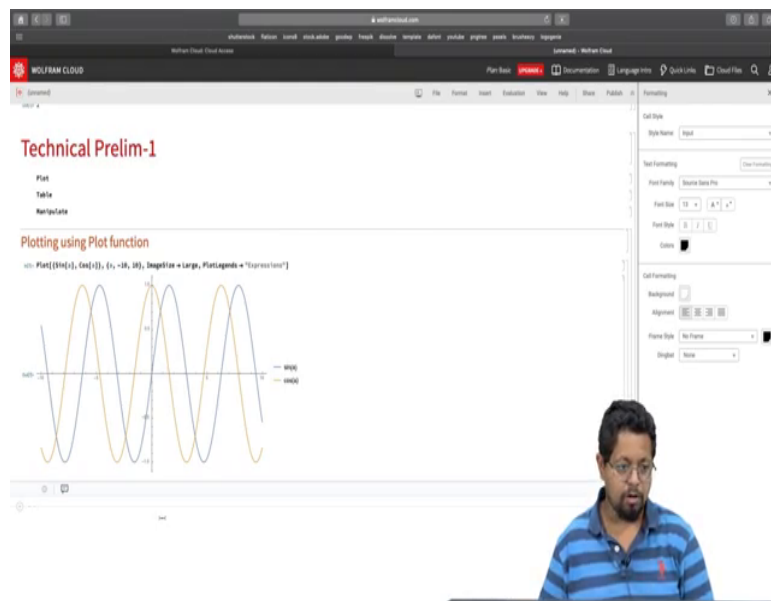
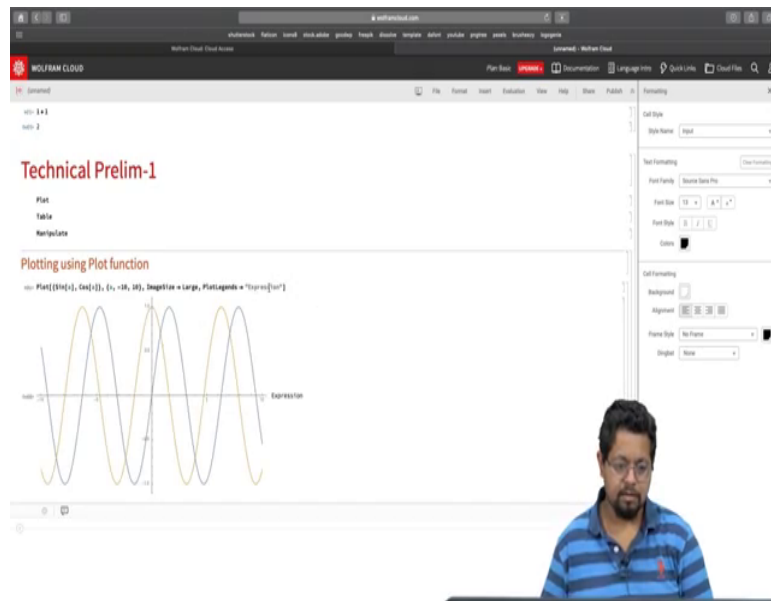
(Refer Slide Time 05:09)



So, I can go ahead and expand the first argument of the Plot function, convert it into a list by enclosing 'Sin[a]' inside curly brackets and then put a comma and say 'Cos[a]'. So, now what I am asking over here is: I am asking Wolfram language to plot 'Sin[a]' and 'Cos[a]' for 'a' between [-10,10]. I will go ahead and press shift enter and I get two plots.

And, as you expect the 'sine' function and 'cosine' function are shifted just by a phase of $\pi/2$. So, you see that 'cosine' function is in orange over here and it peaks at $x=0$ and 'sine' function is vanishing at $x=0$ and sine function peaks at $\pi/2$ which you do not see here, but what we will do is, we will make this plot little bit bigger.

(Refer Slide Time 06:11)



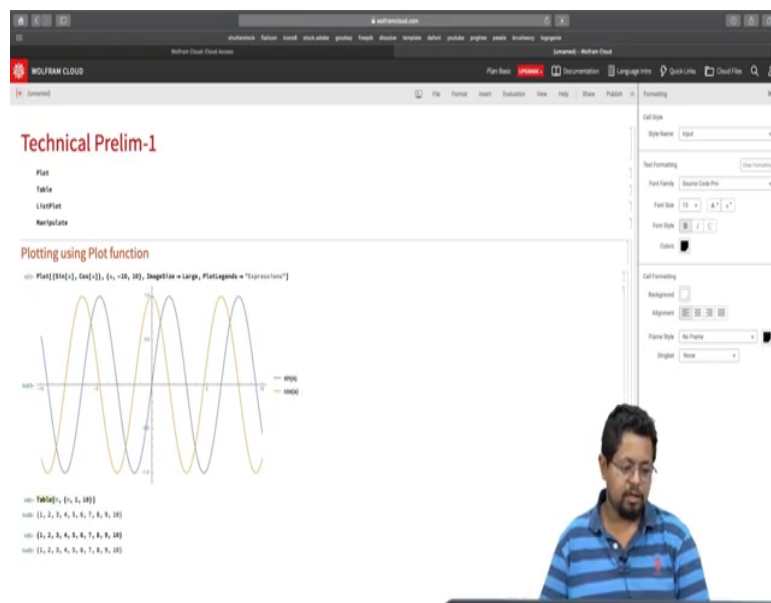
So, you can see that. We will give an option 'ImageSize -> Large'. Now, you can see that this is 5 over here. So, this is approximately $\pi/2$. But, to do that more accurately you can control your ticks and other features of this plot. As we go along, I will show you how to do those things.

But, for now, in order to make sure that blue is 'sine' and orange is 'cosine', I will add another option, which is called 'PlotLegends' and in the curly brackets, I will say this is 'Expression'. When I execute it, it shows me that it adds a legend over here and says that 'Sin[a]' is in blue

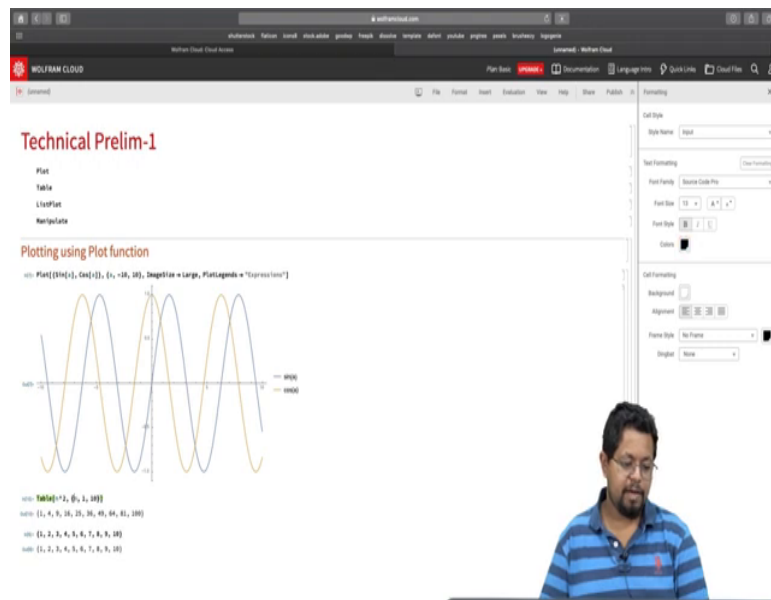
and 'Cos[a]' is in orange. Okay so, this is just a very brief overview, starting with some of the Mathematica functions. As we go along, we are going to learn more and more about this.

Let us go ahead and explore another function called 'ListPlot'. So, let me add that to my list over here, okay, let us go and discuss 'Table' first and then after 'Table' we will go ahead and discuss 'ListPlot'.

(Refer Slide Time 07:54)



The screenshot shows the Mathematica Cloud interface. The main content area displays a plot titled "Plotting using Plot function". The plot shows two trigonometric functions: $\sin(x)$ (blue) and $\cos(x)$ (orange) over the range $x \in [-\pi, \pi]$. The plot is generated using the command `Plot[Sin[x], Cos[x], {x, -Pi, Pi}, ImageSize -> Large, PlotLegends -> "Expressions"]`. Below the plot, the output of the `Table` function is shown for `Table[x, {x, 1, 10}]`, resulting in the list $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. The right-hand side of the interface shows the formatting options for the plot, including style name, font family, font size, and background color.



The screenshot shows the Mathematica Cloud interface. The main content area displays a plot titled "Plotting using Plot function". The plot shows two trigonometric functions: $\sin(x)$ (blue) and $\cos(x)$ (orange) over the range $x \in [-\pi, \pi]$. The plot is generated using the command `Plot[Sin[x], Cos[x], {x, -Pi, Pi}, ImageSize -> Large, PlotLegends -> "Expressions"]`. Below the plot, the output of the `Table` function is shown for `Table[x^2, {x, 1, 10}]`, resulting in the list $\{1, 4, 9, 16, 25, 36, 49, 64, 81, 100\}$. The right-hand side of the interface shows the formatting options for the plot, including style name, font family, font size, and background color.

Now, 'Table' is a very powerful construct or powerful function. It allows you to construct a table of items. So, for example, if I say `Table[n]` and then again in the curly brackets, I say `n`

goes from {1,10}. See what happens. I simply get a list of items over here list of numbers from 1 to 10.

Essentially what this is doing is: I am saying that I want to take n. I want to take this function the first argument of the table and execute it for n = [1,10] and then pack all of that into a list. A list is anything that is between two curly brackets.

So, for example, this is same as 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10. To create this list, I have to type it manually inside curly brackets, or I can create a very large list by simply executing a 'Table' command. You can also think of 'Table' command is something that generates an array of something that you want.

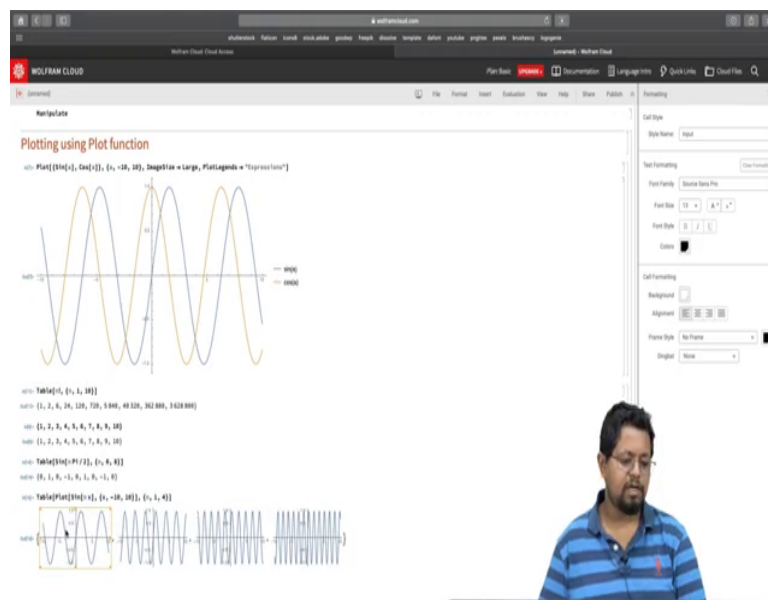
So, let us go ahead and play around with it. Let us go ahead and say I want a table of n^2 . So, I can simply say the first argument of 'Table' function is n^2 and give me the first 10 squares of integers and, as you expect, you are going to get 1, 4, 9, 16, 25 and so on. We can make it more interesting. We can also say I want $n!$. To get a factorial all you have to do is put a factorial sign over here.

maybe put a symbol π . If I do that for the symbol π I again get, because the system recognizes this as π .

Okay, I think it is better to type π like that because pressing the escape key takes me out of the full screen. Let me take another example, I can generate tables of plots. So, inside the 'Table', I will put the 'Plot'. Let us say, `Plot[Sin[n x]]`, should put a space between n and x, because I want to make tables of 'sine' of 1x, 2x, 3x and so on for x lying between [-10,10].

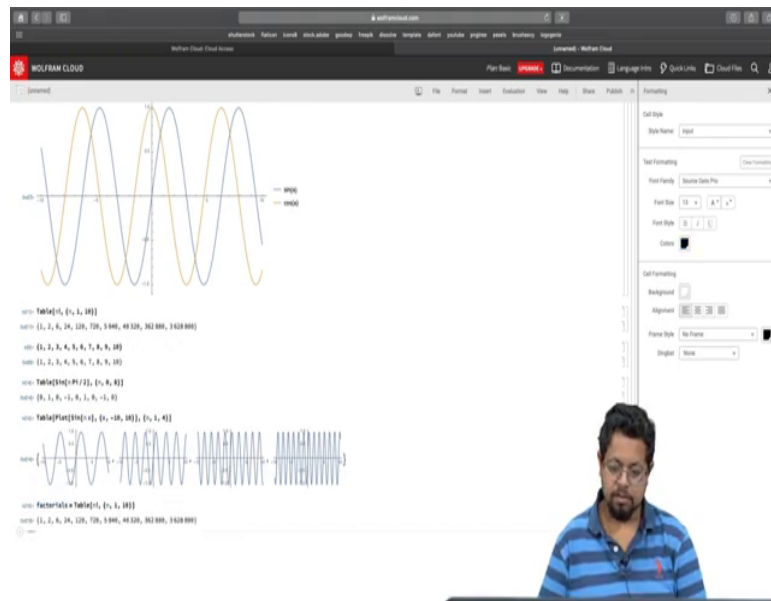
But, I want to take n to go from 1 to 4, so that means its gonna generate me 4 plots. In fact, it is going to be an array of 4 plots. Seems like there is some error. Oh sorry, I forgot to put a comma over here and this should have been x, okay. So, let us go ahead and try this again.

(Refer Slide Time 12:32)



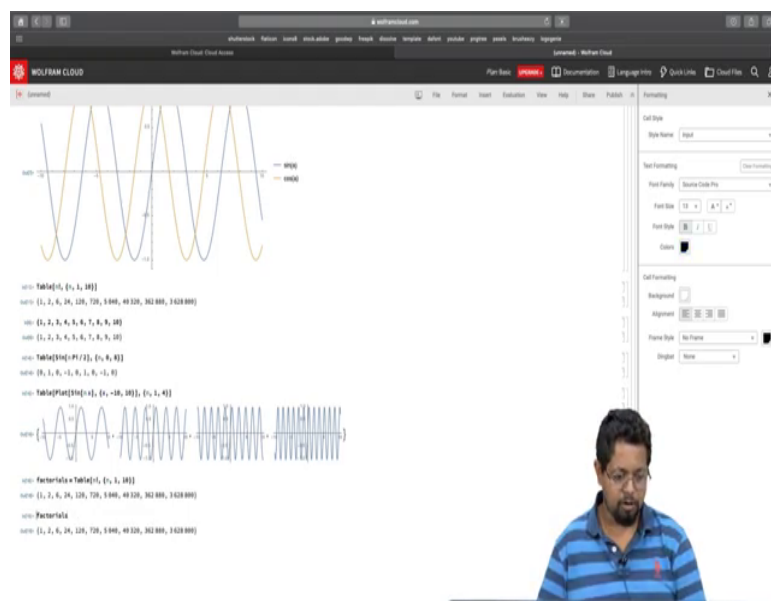
There we go, we have got an array of 4 plots, we can see by clicking on this. This is my first plot for Sin[1x], this is plot of Sin[2x], this plot of Sin[3x], this is plot of Sin[4x] and they are all enclosed in a pair of curly brackets. This is where the curly brackets open, this is where the curly brackets close and I have got 4 different plots. Okay, let us go ahead and go back to n!

(Refer Slide Time 13:10)



So, we have got 'Table' of $n!$, for n from 1 to 10. I will go ahead and assign this array to a variable. This time, for that I will use an equality and I will call my variable to be, let us say, "factorials". So, I am going to assign the variable factorials, the value of the `Table[n!, {n, 1, 10}]` for n from 1 to 10. Notice, that the variable right now is undefined as it is not been assigned any value.

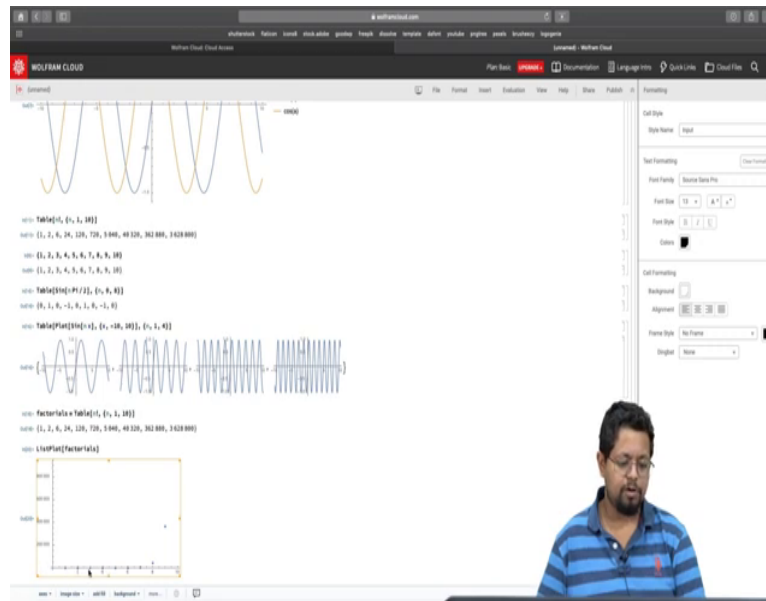
(Refer Slide Time 14:06)



So, it appears in blue the moment I execute this by pressing Shift+Enter. The "factorials" turns into black, which means this variable is recognized by the system next time I actually

type factorials. Not only that it also gives a suggestion, over here I can click on this and then when I execute it, I will just get the same array that the value it has been assigned to. At this point, I would like to make a plot of this.

(Refer Slide Time 14:34)



So, I can say that Plot this but I cannot use the Plot function, because Plot function works for functions whose argument is some variable x but over here, this is a list of items so I will use ListPlot and I will enclose factorials as the array inside the ListPlot. When I make a plot, I get a bunch of points.

You have to look a little bit closely to see the points over here. There is the point here, there is a point here, there is a point here and it appears that most of the points are lying on zero, but look at the y axis, the y axis values is so large.

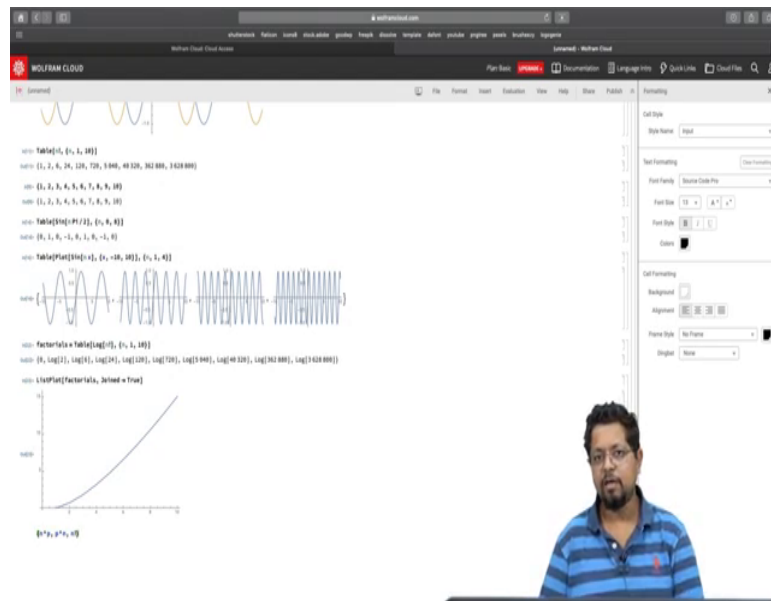
(Refer Slide Time: 15:18)

The screenshot shows the Wolfram Cloud interface. The top plot displays a sine wave with several points marked. Below it, the code `Table[n!, {n, 1, 10}]` is executed, resulting in the list $\{1, 2, 6, 24, 120, 720, 5040, 40320, 362880, 3628800\}$. The next code block `Factorial[n]` produces the same list. The final code block `ListPlot[Factorial, Joined -> True]` generates a plot where the factorial values are connected by a blue line, showing exponential growth. The right-hand side of the interface shows the formatting panel with various options like 'Style Name', 'Font Family', and 'Background'.

Let us go ahead and give an option over here. Let us say 'Joint -> True', which essentially will join all the points. So, now you can see this blue line, which is showing how the $n!$ grows with n . Now, this says $n!$ grows very-very fast with n . You can see from here that for $n=9$, the value corresponding to $n=9$ is much larger than the value corresponding to $n=8$, which is much larger than the value corresponding to $n=7$ and so on.

This is what is expected, you see that in the table over here and this is something not very exciting. So, what we will do is, we are going to take $\text{Log}[n!]$ because \log of $n!$ will not be as large. And when I take $\text{Log}[n!]$, I get this array. Mathematica decides to simply write this log of that, but does not really matter to us, what we will do now is: we will make a plot.

(Refer Slide Time 16:16)



And now, you see the $\text{Log}[n!]$ actually goes much more slowly. The y-axis is not as big and now, you can actually think of comparing this quantity with something. As a tiny exercise, let us think about comparing the following things. We will compare 2^n , e^n , sorry let us say n^p where p is a number, e^n which is an exponential.

Or, let us say p^n and $n!$. Which one of these do you think grows the largest or grow the fastest, as we increase n ? This is something we can simply test out by plotting. So, what we will do is: we will generate three arrays: one for the factorials, one for the polynomials and n^p is a polynomial, p^n is an exponential and $n!$ is factorial. So, what we will do is: we will generate our first array "polynomials" for a fixed value of p .

(Refer Slide Time 17:48)

The image displays two screenshots of the Wolfram Cloud interface. The top screenshot shows a plot of a polynomial function, $y = x^4$, and a list of values for $\text{Table}[n^4, \{n, 1, 10\}]$. The bottom screenshot shows a plot of a logarithmic function, $y = \log(x)$, and a list of values for $\text{Table}[\log[n], \{n, 1, 10\}]$. A person is visible in the background of both screenshots.

So, for that I will say $\text{Table}[n^p]$ and just as an example, we will choose $p=4$ and I will say n goes from 1 to 10 and we will go ahead and execute that. So, this will generate one array which I have named polynomials and this is what the numbers look like.

Next, we will go ahead and, say exponentials. If you want, we can copy this from here or we can simply go ahead and say this time 4^n . So, n is in the exponent now, and I will say n goes from 1 to 10, as before. And now, you see we get another sequence of numbers and these numbers - initially they are somewhat smaller than these numbers, but eventually, they become and become much larger.

And finally we are going to take factorials here and we will say $n!$ and n goes from 1 to 10. And that will generate me the factorial sequence. Now, I want to make a ListPlot of these simultaneously. For that, in a curly bracket, I will include all three of them, which is the way to make a ListPlot of all of them on the same plot.

Polynomials, exponentials and factorials, for some reason this is in some strange blue colour but that does not matter, we will go ahead and plot it and you see there are dots of three different colours which is still hard to see.

(Refer Slide Time: 20:02)

```

ListPlot[Factorials, Joined -> True]

(* *) a^n, n! = 0)
polynomial = Table[x^4, {x, 1, 10}]
exponential = Table[4^x, {x, 1, 10}]
factorials = Table[x!, {x, 1, 10}]
ListPlot[{polynomial, exponential, factorials}, Joined -> True, PlotLegends -> {"polynomial", "exponential", "factorial"}]

```

```

ListPlot[{polynomial, exponential, factorials}, Joined -> True, PlotLegends -> All]

```

So, I will say that 'Joint -> True'. And to understand which colour corresponds to what, I will also go ahead and add PlotLegends option. So, in order to create this arrow, I press a dash and then a greater than sign and that creates an arrow and in the quotes, I will go ahead and type Expressions which means that give me a legend for this plot, with expressions or that would not work, because there is no expression for this.

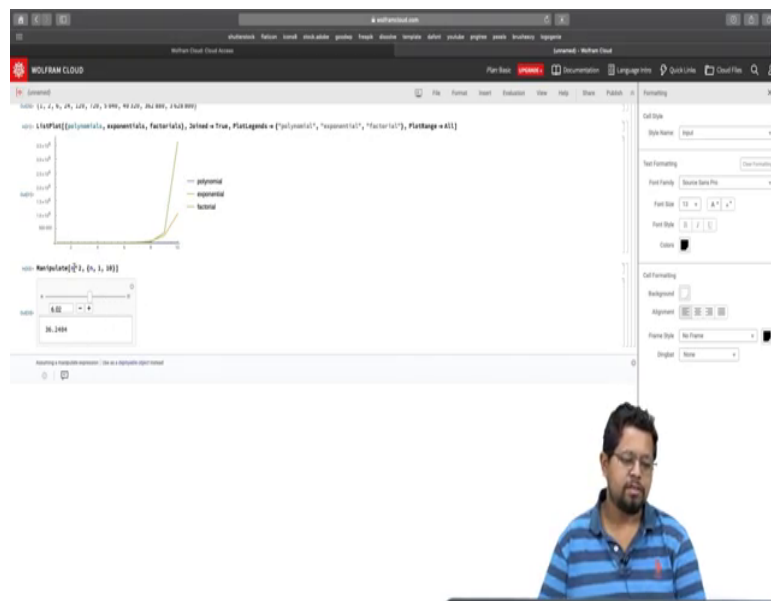
This is simply data. So, we will say that PlotLegends are: the first one was polynomial, the second one that we use in the array over here is exponential, and the third one is factorial. So, blue line is for polynomial, which is the slowest growing function.

The orange line is for exponential which appears to grow really fast. And then finally third one is factorial, which eventually picks up and goes very fast. Now, since I am not plotting everything, it is unclear what is happening, how they are competing because I have not plotted what is happening beyond 8 for these two functions.

So, to get that, I will add another option called PlotRange and I will assign it a value "All". Let us see what happens with that, there we go. So, polynomial is pretty much invisible because polynomial growth is very, very slow. You are comparing this 10000 with, this 3 million.

And then exponential is the next one, and eventually, the factorial picks up. Somewhere, in the beginning, exponential was leading the factorial and then eventually factorial growth picks up and becomes faster than all these functions. So, this was a quick example of, how we can compare these various functions. Let us go ahead now, and look at the last item in our list today the 'Manipulate' function.

(Refer Slide Time 22:41)



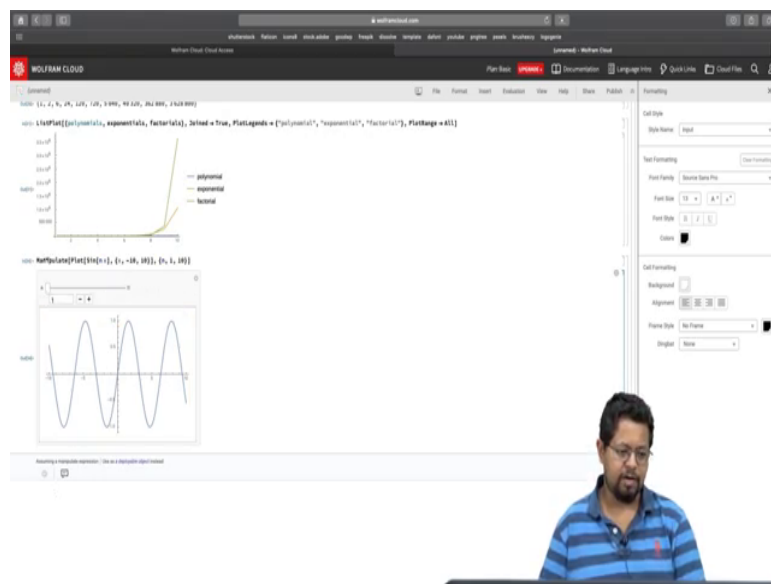
The Manipulate function turns out to be something very useful. It allows you to control a parameter and vary it by moving the sliders. So, suppose I say n and for the first argument of Manipulate, and for second one I say {n,1,10}. Let us go ahead and execute and see what happens.

Yeah okay, so now you see the output of Manipulate is this frame, where I see a slider marked with n and a value printed over here 1, which is the value of n . If I click on this little tiny plus button over here, it shows me the value of n , and shows me a plus and minus sign which says, if I click on this, it will increase n , and as a consequence, the value of n will change here and correspondingly the value of n will change here.

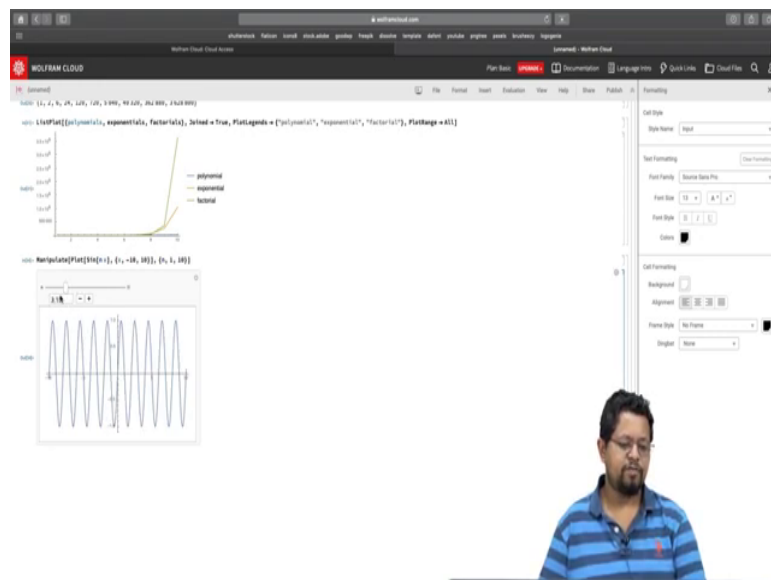
This is the value of the parameter and this is the value of the display. Right now, the parameter that we are varying is also the value of the function that you want to display. So, I am getting the same value over here. Let us, go ahead and make this n^2 , execute it. I can also go ahead and just move this slider, and as I move this slider you can see this number changes. I can click somewhere on the slider.

The value of n is 3.82 and you get 14.59 corresponding to that, for n^2 . I change this value of n to 6.02, I get a square of that is 36.24 and so on. Well, this is not very exciting.

(Refer Slide Time 24:49)



The screenshot shows the Wolfram Cloud interface. The top part of the interface displays a plot with three curves: a polynomial (blue), an exponential function (orange), and a factorial function (green). Below this, there is a Manipulate slider for a sine wave plot, $\text{Plot}[\text{Sin}[n x], \{x, -10, 10\}]$. The slider is currently set to 1. The right side of the interface shows the formatting options for the plot, including Call Style, Style Name, Font Family, Font Size, Font Style, Colors, Call Formatting, Background, Alignment, Frame Style, and Display.



The screenshot shows the Wolfram Cloud interface. The top part of the interface displays a plot of $\text{Sin}[n x]$ for x between $[-10, 10]$. Below this, there is a Manipulate slider for changing the value of n . The slider is currently set to 1. The right side of the interface shows the formatting options for the plot, including Call Style, Style Name, Font Family, Font Size, Font Style, Colors, Call Formatting, Background, Alignment, Frame Style, and Display.

What we will do is, we will replace this function with a Plot of $\text{Sin}[n x]$, for x between $[-10, 10]$. Let us go ahead and execute that. This time the output of Manipulate is a plot, which is inside this frame along with the slider for changing the value of n . Since, the first value of n is 1 here, the slider is initialized, the value of n is initialized by 1 and I am getting a plot of $\text{Sin}[1 x]$.

I never said what is the value of n I wanted over here, but manipulate automatically initializes the first value that is given for n and when I do that, I can now go ahead and slide the slider. As, I slide the slider the value of n gets updated and the plot gets updated.

(Refer Slide Time: 25:48)

The screenshot shows the Wolfram Cloud interface. The top part of the interface displays the code: `ListPlot[{polynomial, exponential, factorial}, Default = True, PlotLegends -> {"polynomial", "exponential", "factorial"}, PlotRange -> All]`. Below the code is a plot showing three curves: a polynomial (blue), an exponential (orange), and a factorial (green). The bottom part of the interface shows the code: `Manipulate[Plot[Sin[x], {x, -10, 10}], {n, 1, 10}]`. Below this code is a plot of a sine wave with a period of approximately 20 units. The right side of the interface shows the formatting options for the plot.

The screenshot shows the Wolfram Cloud interface. The top part of the interface displays the code: `ListPlot[{polynomial, exponential, factorial}, Default = True, PlotLegends -> {"polynomial", "exponential", "factorial"}, PlotRange -> All]`. Below the code is a plot showing three curves: a polynomial (blue), an exponential (orange), and a factorial (green). The bottom part of the interface shows the code: `Manipulate[Plot[Sin[x], {x, -10, 10}], {n, 1, 10}]`. Below this code is a plot of a sine wave with a period of approximately 2.8 units, indicating a higher frequency than the previous plot. The right side of the interface shows the formatting options for the plot.

So, what happens when I change n, I am essentially increasing the frequency of 'sine' function and the 'sine' functions is oscillating more and more rapidly. Very well, so this was a little bit about Manipulate function.

(Refer Slide Time: 26:05)

The screenshot shows the Mathematica Cloud interface. The main window displays a plot with three curves: a blue line for 'polynomial', a red line for 'exponential', and a green line for 'factorial'. The x-axis ranges from 0 to 10, and the y-axis ranges from 0 to 1000. The exponential function is significantly higher than the polynomial, and the factorial function is the highest. Below the plot is a slider control for 'n', currently set to 3. The code in the input field is: `ListPlot[{{polynomial, exponential, factorial}, Default = True, PlotLegends -> {polynomial, exponential, factorial}, PlotRange -> All}]`. The right-hand side shows the 'Formatting' panel with various options for the plot.

This screenshot is similar to the one above, but the slider for 'n' is now set to 4. The plot shows the same three functions, but the exponential and factorial functions are much higher, demonstrating their rapid growth compared to the polynomial. The code in the input field is: `ListPlot[{{polynomial, exponential, factorial}, Default = True, PlotLegends -> {polynomial, exponential, factorial}, PlotRange -> All}]`. The right-hand side shows the 'Formatting' panel.

If you want to initialize n by different value let us say 3, so, when you do that, now this will allow n to be initialized by 3 and you see the slider is set at $n = 3$, click on the small "+" button to see that.

Now as you change the slider, you can change the value of n and with that the corresponding plot of the function. This was just some prelims to get you to understand how some of these functions work in Mathematica, and in our next lecture, we will use all these constructs.