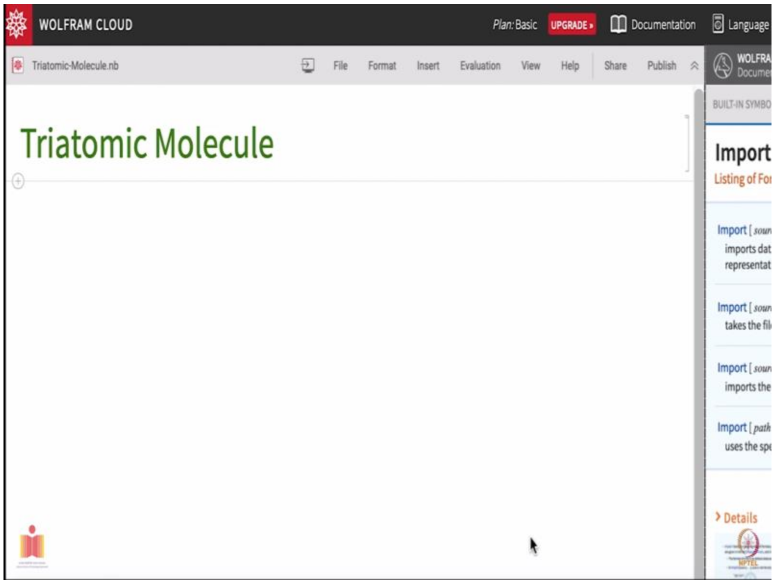
Introduction to Classical Mechanics
Professor. Anurag Tripathi
Assistant Professor
Indian Institute of Technology Hyderabad
Lecture 67
Normal modes of Triatomic Molecule using Mathematica

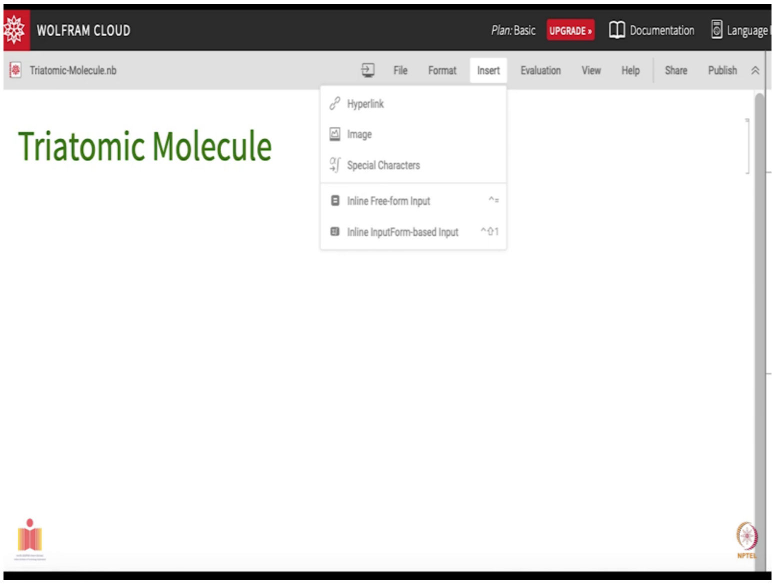
(Refer Slide Time: 00:14)



In this video, I would like to revisit the calculation of Triatomic Molecule and this time I want to show how to do it using Mathematica. However, you may use whichever programming language you prefer to use, for example, Python or any other one. This is the one which I know and I use. So, I am going to describe the competition using this one and I hope after this, I would have encouraged you enough to or persuaded you enough to use certain computational tools to also do the competitions or check the competitions.

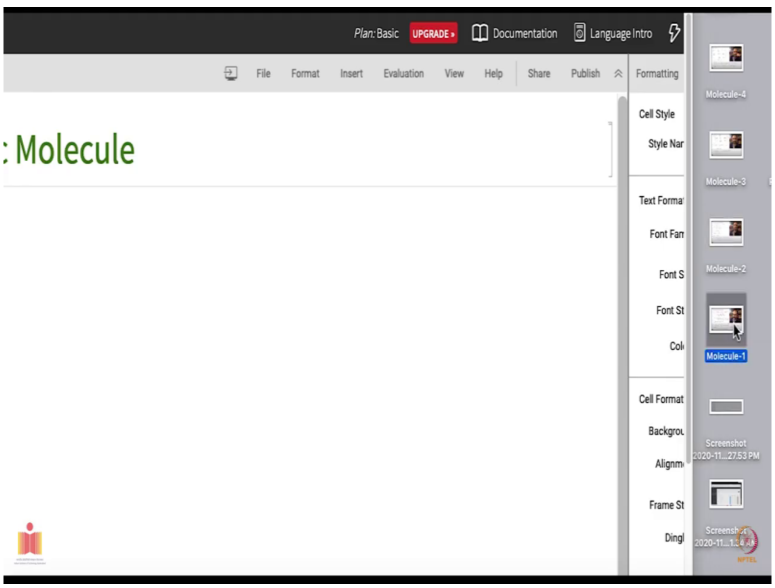
And these are also very useful if you are trying to give a lesson in the class. So, let us begin now what I will do is I will start by bringing in some images of the lecture which I gave. So, I have put them on my desktop and I am going to insert it here.

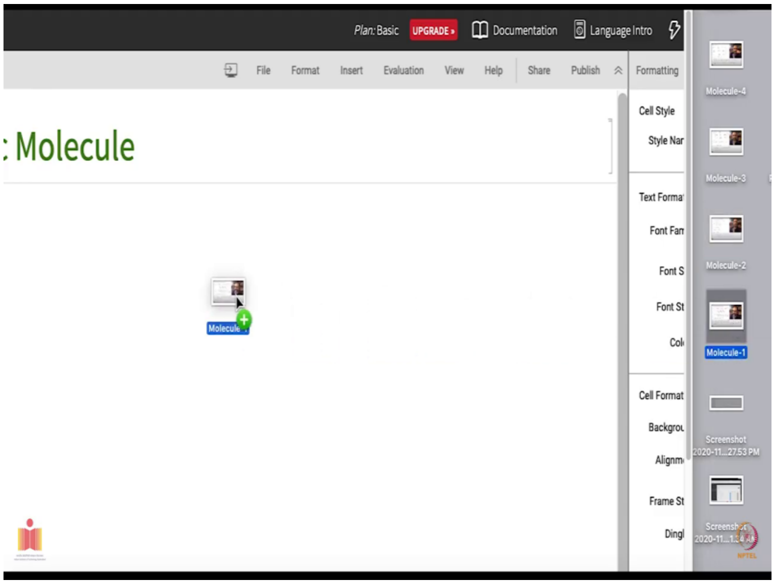
(Refer Slide Time: 1:22)

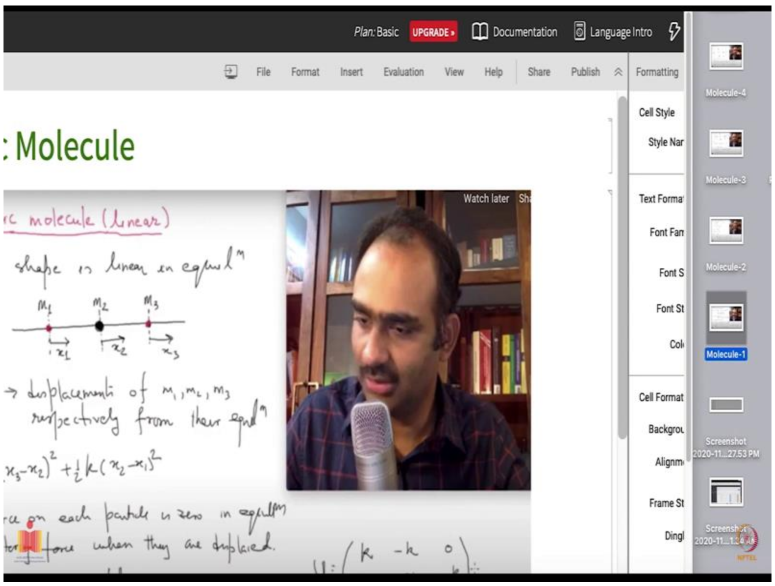


So, there is a way to insert, no not here, insert. So either I can provide the path or I can directly just I will show you how to do it.

(Refer Slide Time: 01:42)

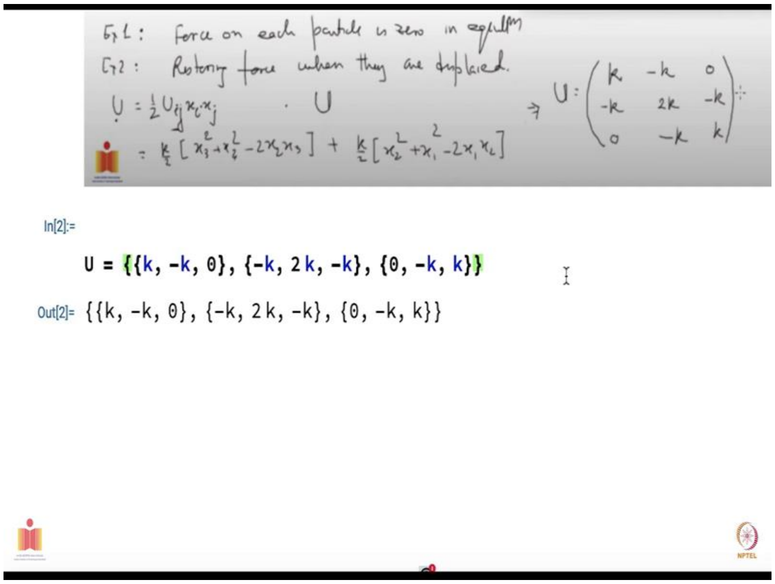






So we go here. This is one of the files. I bring it here. So, see it is more like a document to me now, where not only I have calculation, but I have for example the notes. So, this is what we were talking about Triatomic molecule earlier and here is the matrix U.

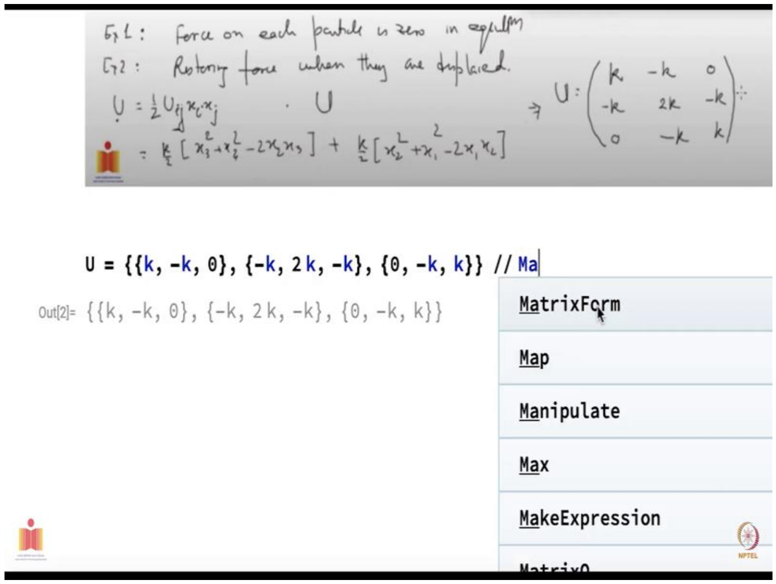
(Refer Slide Time: 02:21)

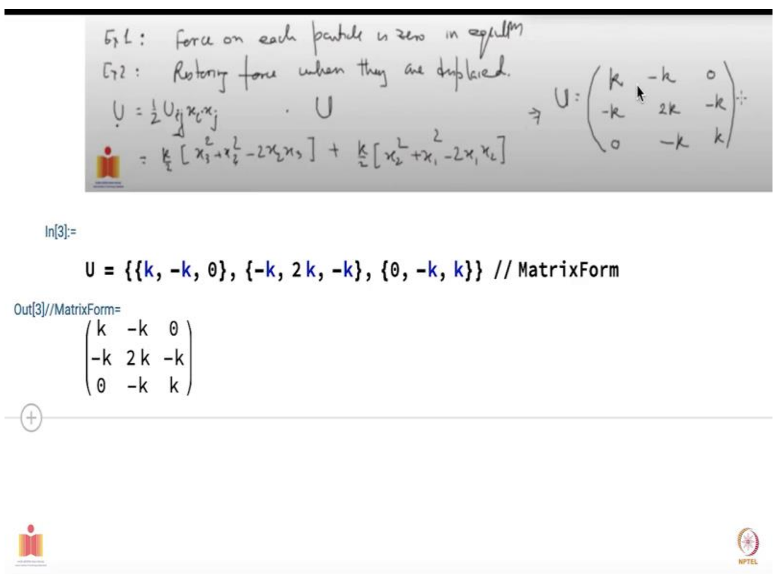


Let me input this matrix, I should come out of the cell and let me define matrix U. To be have to input an array or matrix, which is also an array. So I start, (())(02:28) need to have three rows or three columns, I put it like this.

So, first row is k minus k 0, k minus k, 0. Second one is minus k, 2 k, minus k and third one is 0, minus k, k. I am not sure whether you can see it nicely, so I can try to do formatting, it appears to be slow. Let us see. Maybe I should select first. Nothing works. So I do not know, so I can show you this way. So, this is what I have input here. Now let us shift enter.

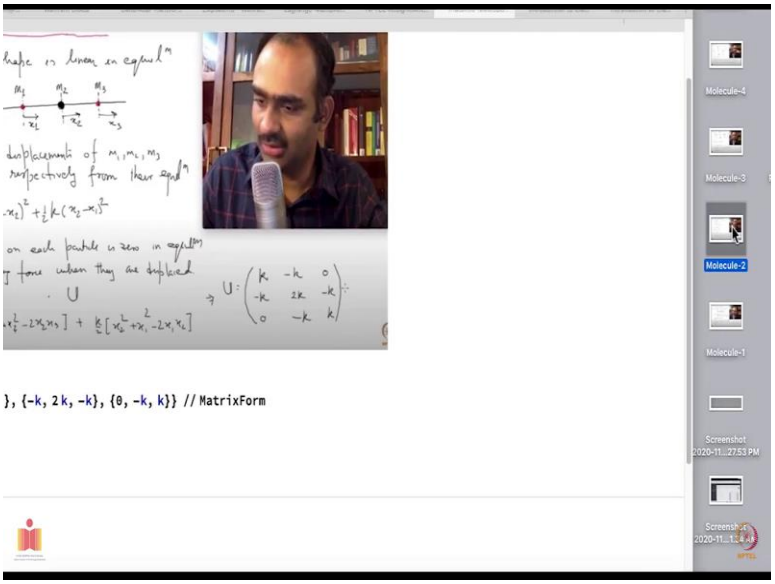
(Refer Slide Time: 03:53)

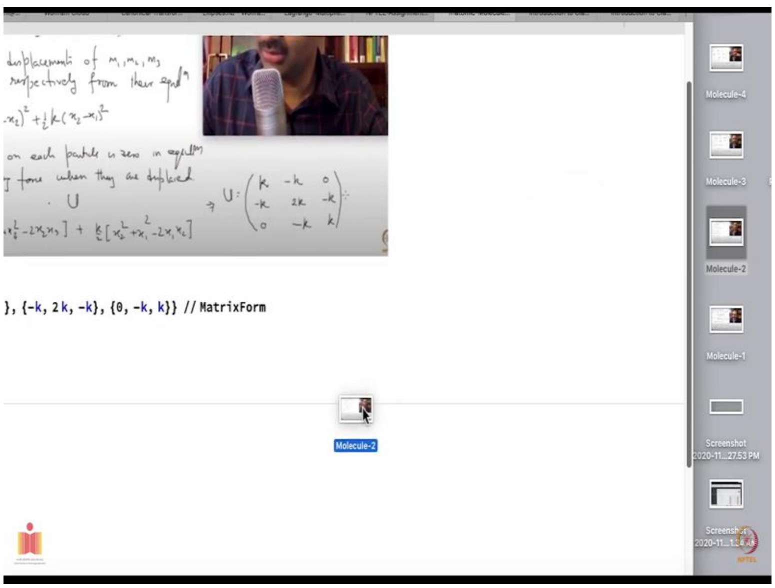


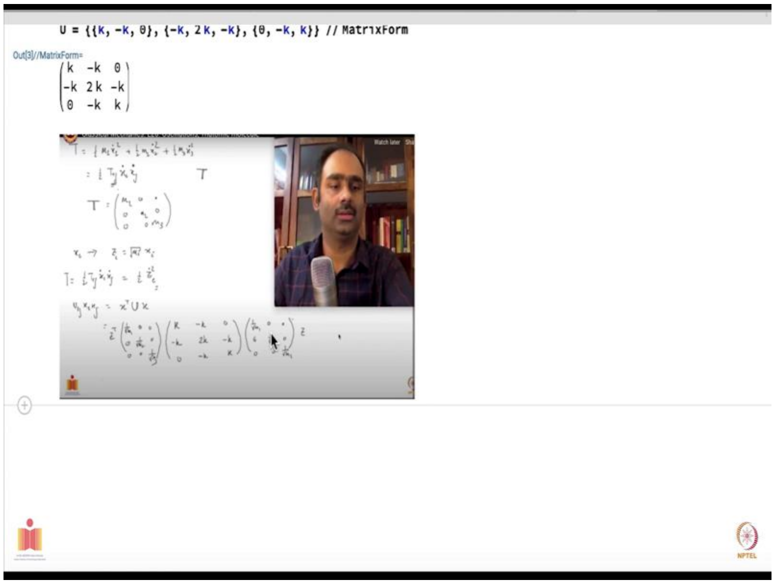


Let us see whether all good by putting it in matrix form. It automatically shows you the option so I have put it like this here and now I can compare with what I have here in the slide. k, minus k, 0 minus k, 2 k, minus k, 0 minus k, k. So this is fine.

(Refer Slide Time: 04:34)

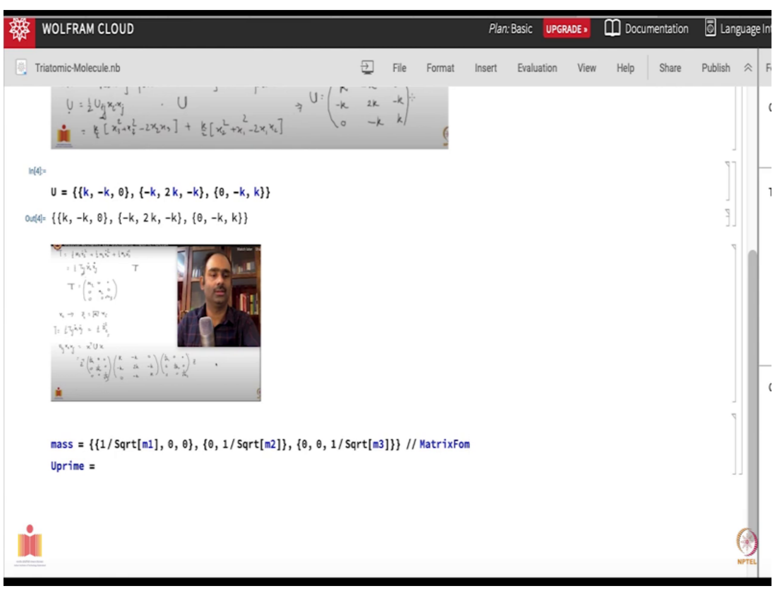


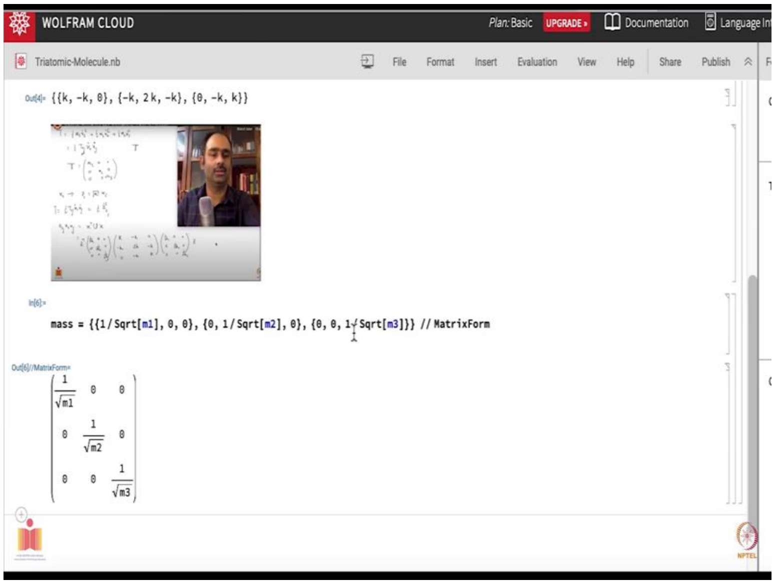




Now let me go back and pull out the other Molecule 2. So, I will take it here let us see. Now this has is too big. Now, if you see here you will recall that now I have to multiply the matrix U, which is here in the center with these two diagonal matrices which contains the one over square root 2, square root of the masses. So, let me do that now.

(Refer Slide Time: 05:34)

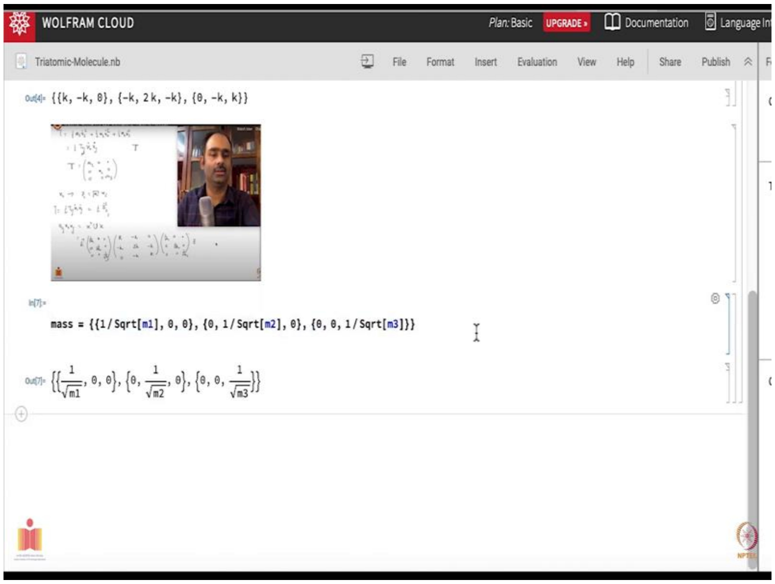


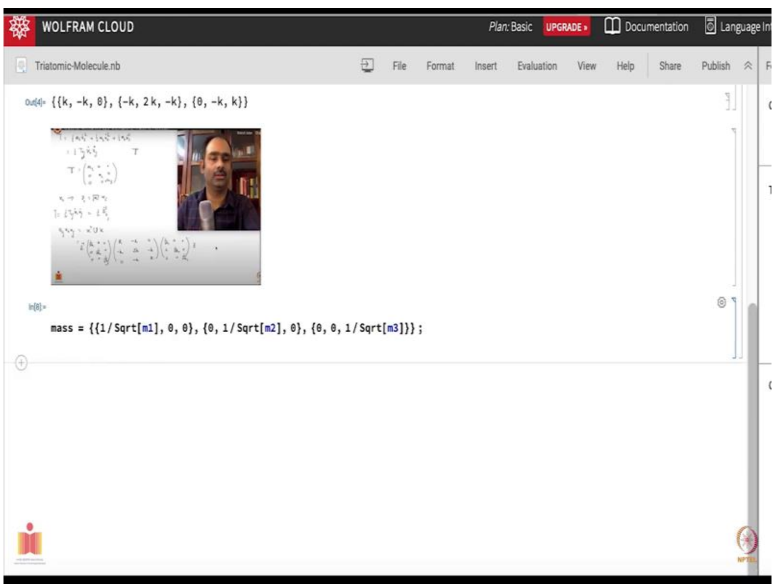


So, here I will remove a Matrix form, otherwise it will not manipulate, so I shift enter again. So, that it is the correct expression is loaded. Now, I define U prime, so remember this product of these three matrices is what we were calling U prime and so I say U prime is equal to before I do that let me first define this matrix let us call it mass, mass matrix.

Which was defined to be this, so what is that here if you see 1 over square root of m1 this space here in the, in the image here as I define 1 over Sqrt and m1 0 0 0 1 over Sqrt and m2 0 0 1 over square root m3 reduce the size of the that m3. Let us check, by putting MatrixForm whether I have inputted currently, something is, I mean of course, this is wrong here. This one should have worked. No, that is having a problem here. I should put a 0. Now, this looks perfect.

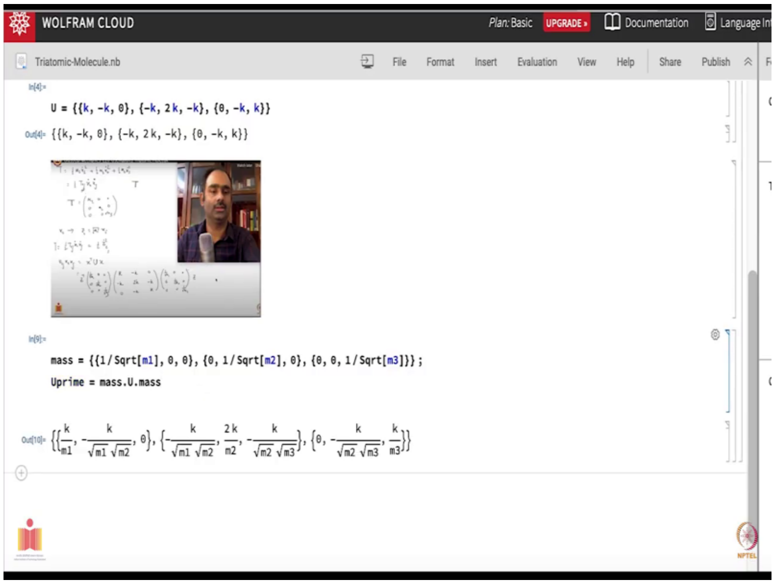
(Refer Slide Time: 07:52)

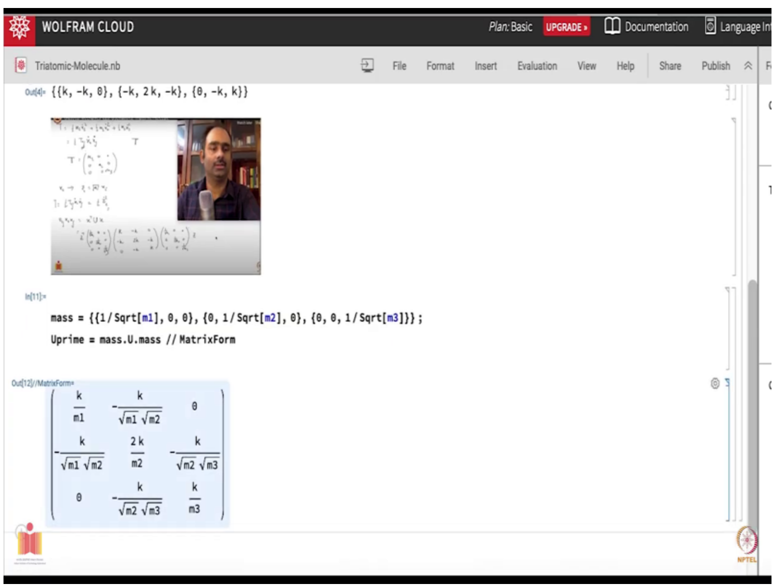




So, I again as before I remove this and run it again and if I want to hide it, I can just put a semicolon and then run it this will not show the result.

(Refer Slide Time: 08:04)

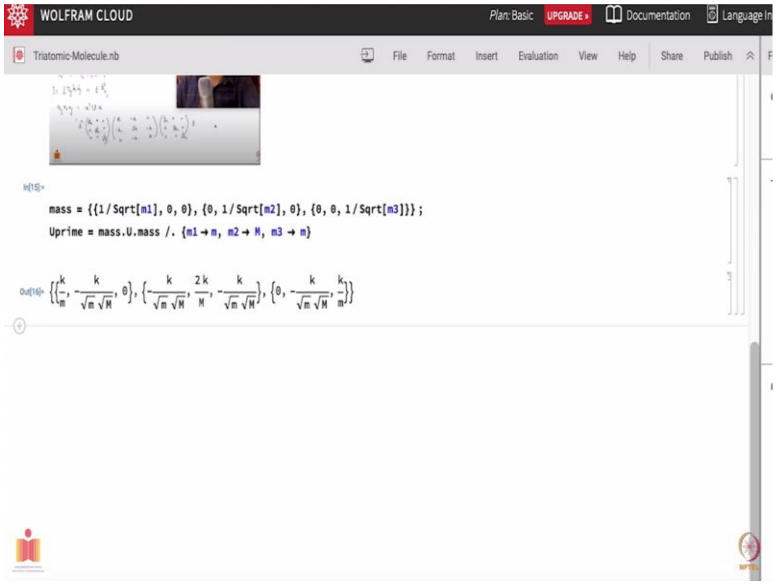


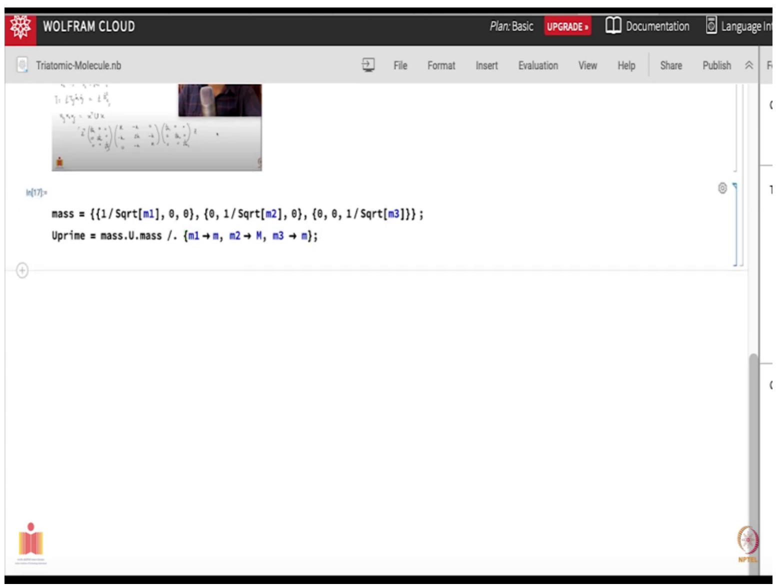


So, I have this matrix and to construct U prime I should write U prime, prime is equal to mass, I think it should be a dot yes dot and U times mass.

You do not need to create a transpose because this is a diagonal matrix. So, that is no problem. Shift Enter and let us see what we get again to see the result clearly MatrixForm. That is the matrix which we have, let me see if I have my older note. Anyway. so that is the m prime, U prime. Now, in the lecture I had identified m1 and m3 to be small m and m 2 to be capital M. So, that is what I am going to do now.

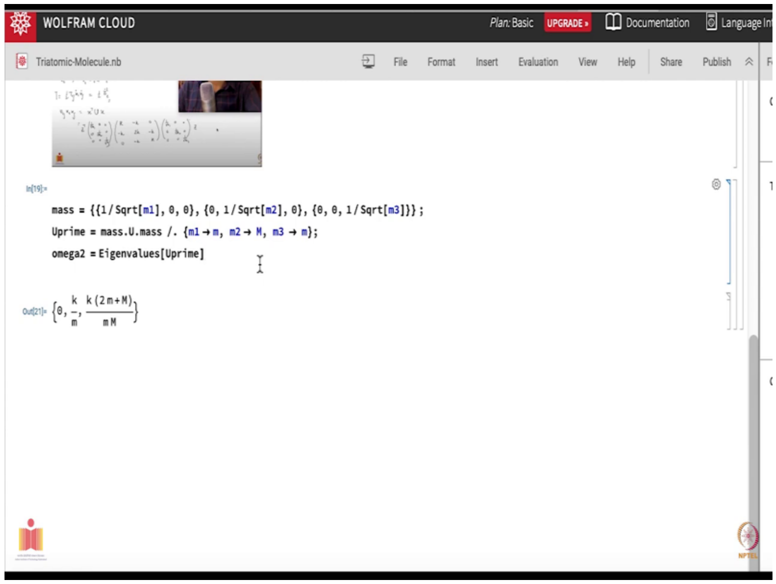
(Refer Slide Time: 09:25)





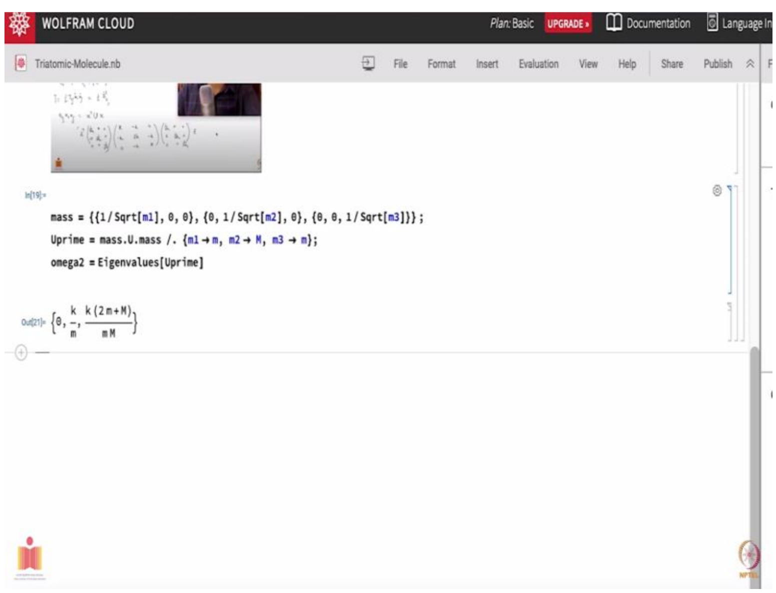
So here, let me remove this again. That purpose, it has solved the purpose, let me put slash dot meaning I am going to replace and m1 is replaced by small m and m2 the middle atom has a mass capital m and the third one as mass m small m. So, these are the replacement rules I have given and let us see, that is nice. Everything looks fine, that is that is good. So, I put a semicolon here, so that I do not see the output. Now, this is the U prime and if you recall this is the matrix which gives us the normal frequencies, the Eigen frequencies of the normal modes.

(Refer Slide Time: 10:50)



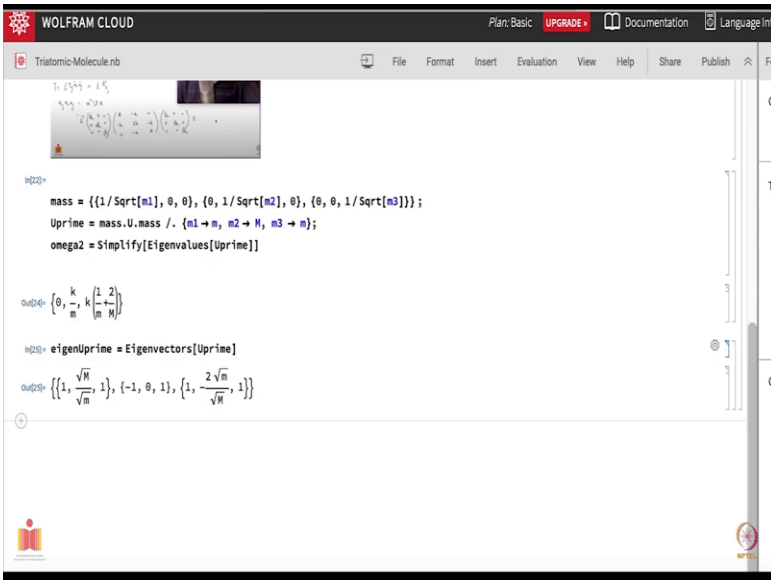
So, I should find out the Eigen frequencies here and for that I just look at the Eigen values selected this of U prime that will give us the omega square omega 2 which the 2 is to remind us that we are looking in the squares. Let us see what happens, it gives the result nicely. That is very nice, what is that, it that happened because others had to struggle to simplify but anyway, my students in the class. Anyway, so this is the, these are the omega squares and if you take the square roots you will get the frequencies.

(Refer Slide Time: 11:51)



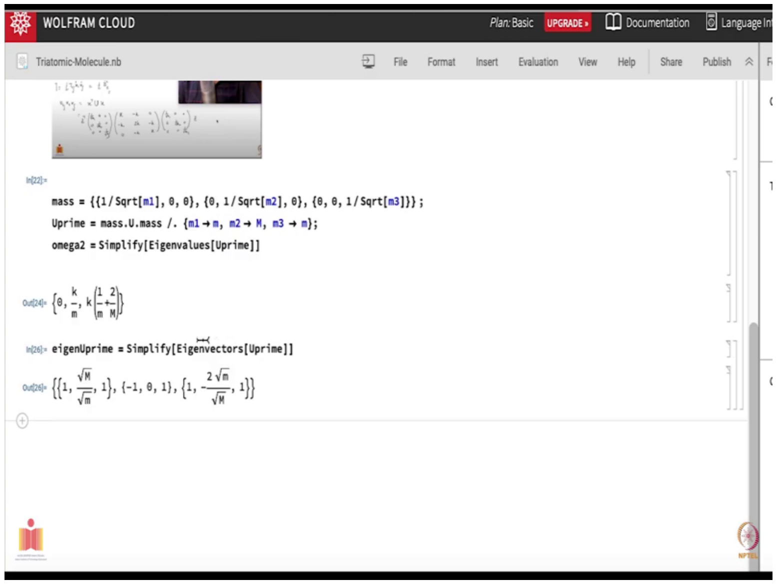
Let me just try to do reduce simplify, let me simplify here. See if something happens, this is more familiar, this is what we have already seen in the lecture. So, you get the frequency here, let us, let us keep it that way and not hide it. So, as far as the frequencies are concerned, this is the result. Now, I go to another block here and now I want to find the normal coordinates and for that I will have to find the Eigen vectors of U prime.

(Refer Slide Time: 12:50)



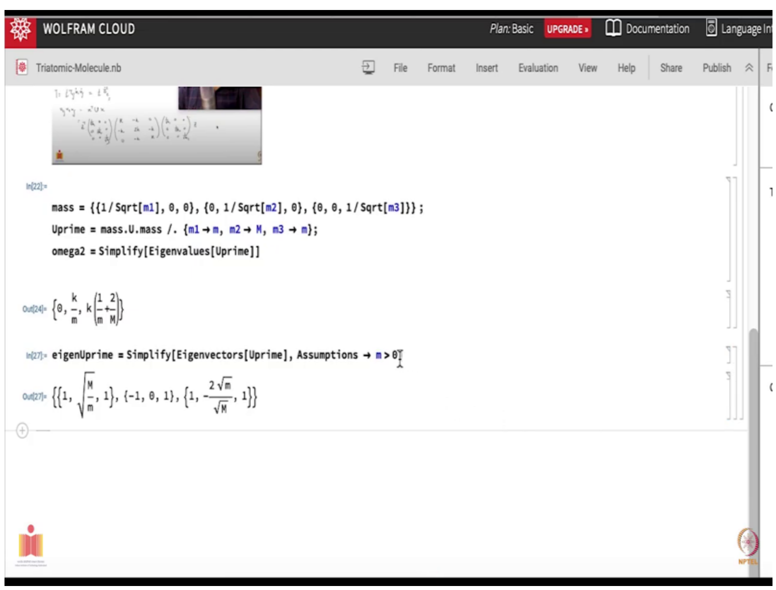
So, Eigen vectors it is built in. So, it is very nice that I do not have to write a small program or something to create to obtain the Eigen vectors that it can Mathematica can do it for us. So, it is nice. I write U prime. That is all that is all that I should do. Eigen vectors. Eigen U prime. That is the name I am giving this to this expression. Let us see, nice. It gives something. That is good.

(Refer Slide Time: 13:51)



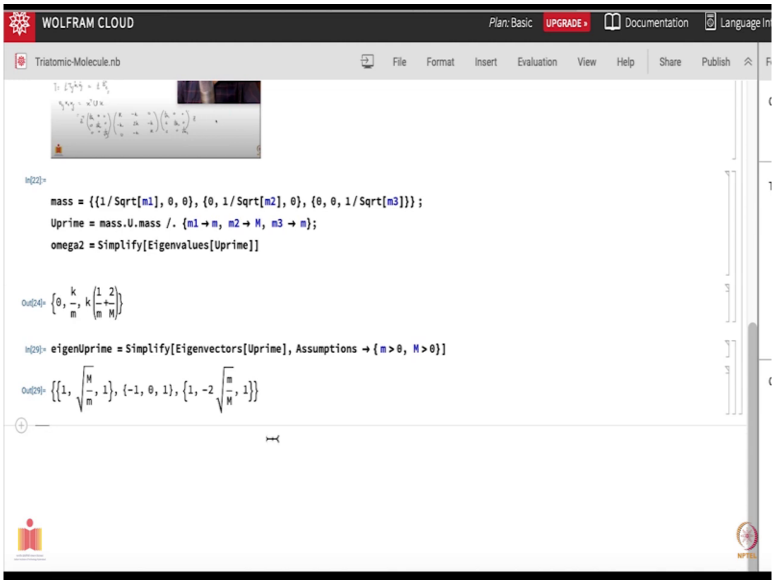
Let me see if I put a simplify whether it can put the these two capital M and small m in 1 square root, simplify. Does it? No, no.

(Refer Slide Time: 14:10)



We can try again. We can say that, I am assuming. See, Mathematica is always trying to be very careful. It is assuming that the capital M and small m are complex. It is not taking chance and it is saying maybe they are complex, so we tell it that, no you can relax a bit. Because the small m is greater than 0 and also, this one will work. Let us see. Good.

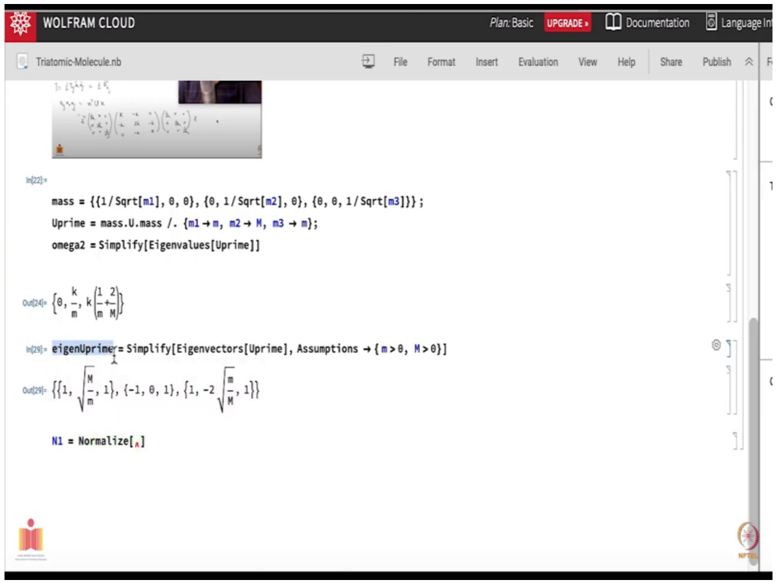
(Refer Slide Time: 14:54)

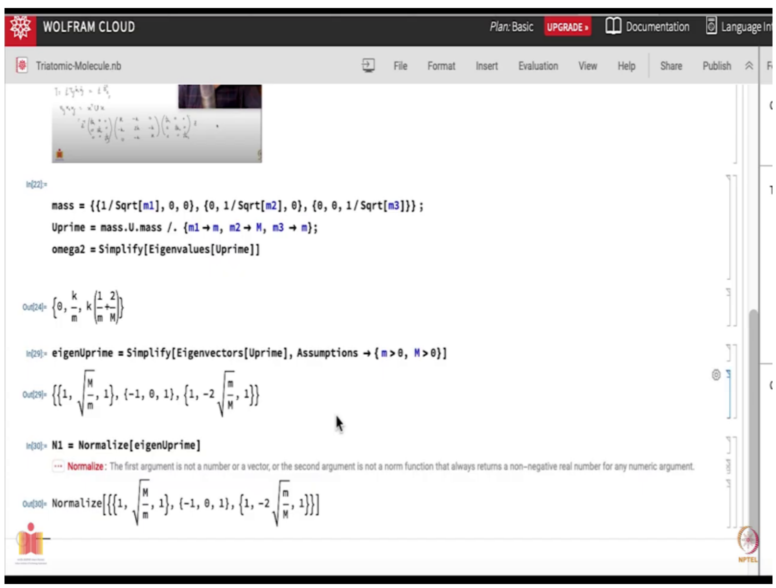


So maybe capital M also if I put. So, I have done something wrong, probably I should have done this yes. Now this is good. So, this is correct. So, I have found the Eigen values of the matrix U prime. Now, if I want to diagonalize the matrix, I should construct the diagonalizing matrix.

For that I have to take all these Eigen vectors and make the matrix O by putting these as column vectors. So, that is what I am going to do, but I have to remember that I should put only the normalized Eigen vectors, if they are not normalized, I am not going to get the orthogonal matrix so. So, let me first normalize this and nice thing is that Mathematica has this function also has built in functions.

(Refer Slide Time: 16:22)

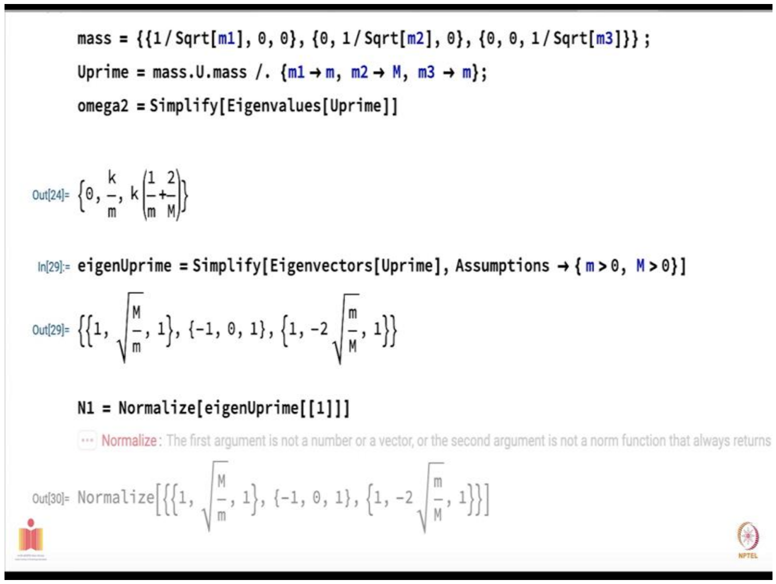




So, I just put Normalize and what should I normalize, I should normalize E I G E N, it is better if I copy paste this thing. Otherwise, I may introduce a, let us see whether it works. I think it will not work. Let us see. It does not work. So, it says the first argument I hope you can see, let me I am sorry, I did not enlarge it.

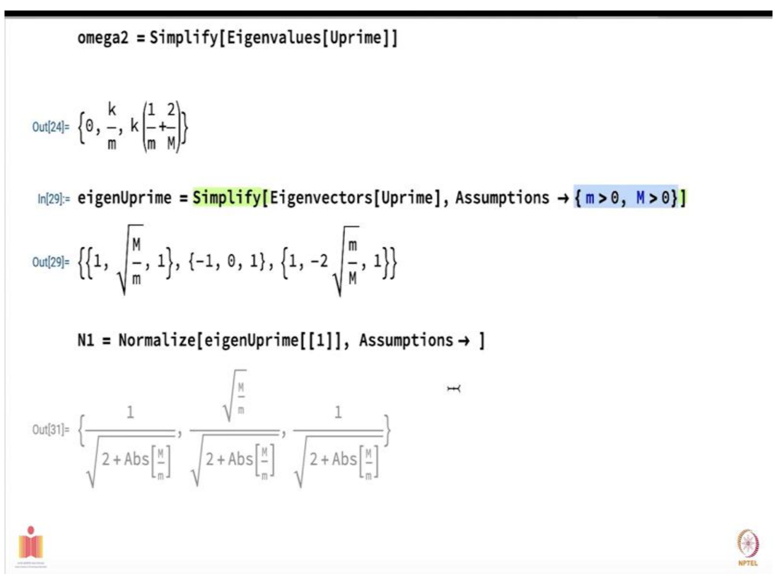
Now you can see whatever I have done till now. So, it is saying the first argument is not a number or a vector, which is true because I have given an array rather than a vector. So, what I should do is say here these are three vectors, one vector, two vector and third vector here.

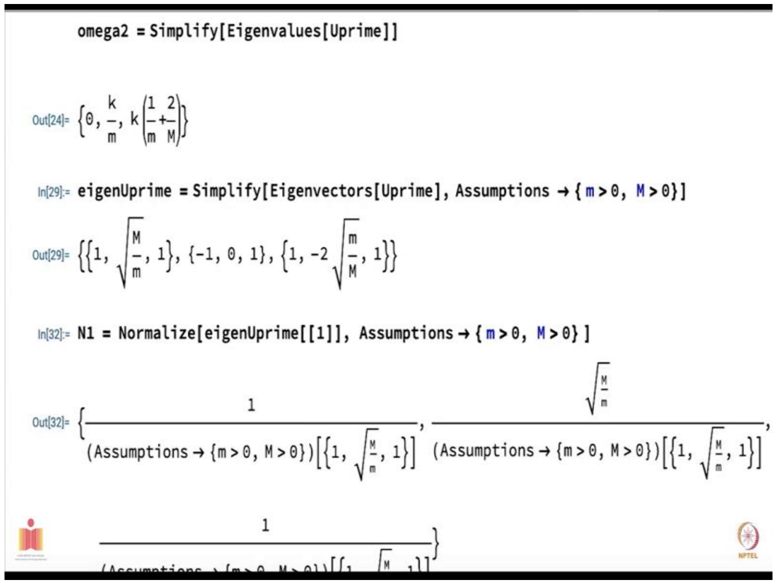
(Refer Slide Time: 17:17)



So what I should do is I should select one of the vectors and that I do by doing let me we check yes it is absolutely correct, let us say the first entry of the array. If I do this, it should give an answer. Again, Mathematica is trying to be very careful and we can again tell it to be a bit relaxed by telling, let me check.

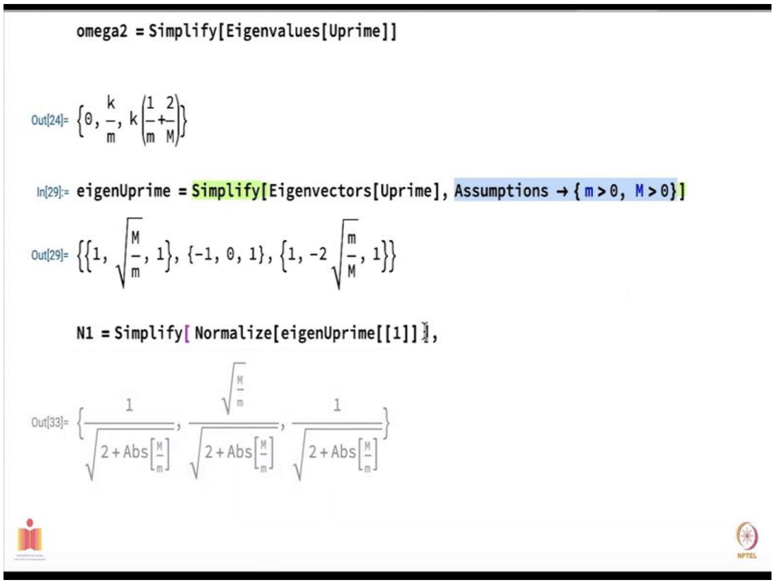
(Refer Slide Time: 17:51)

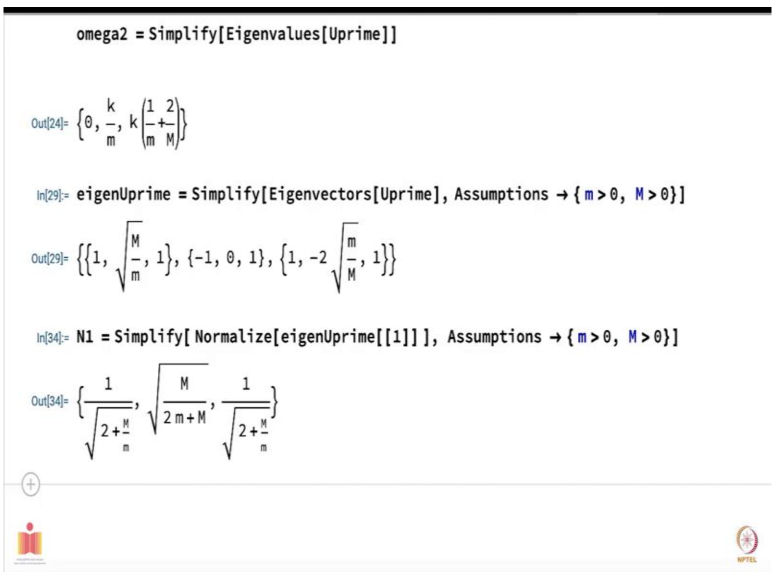




I do not know whether this will work let us see assumptions. Again, the same thing. Copy and paste here, let us see whether that works. No it does not work.

(Refer Slide Time: 18:12)

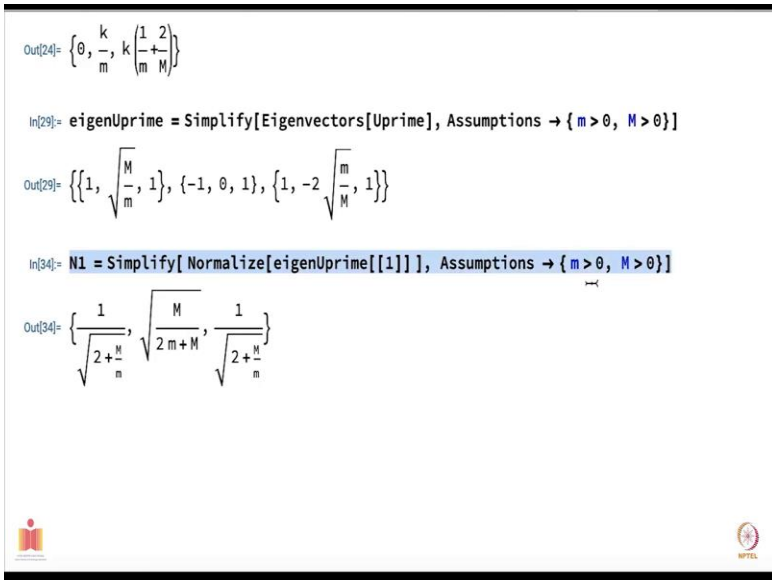


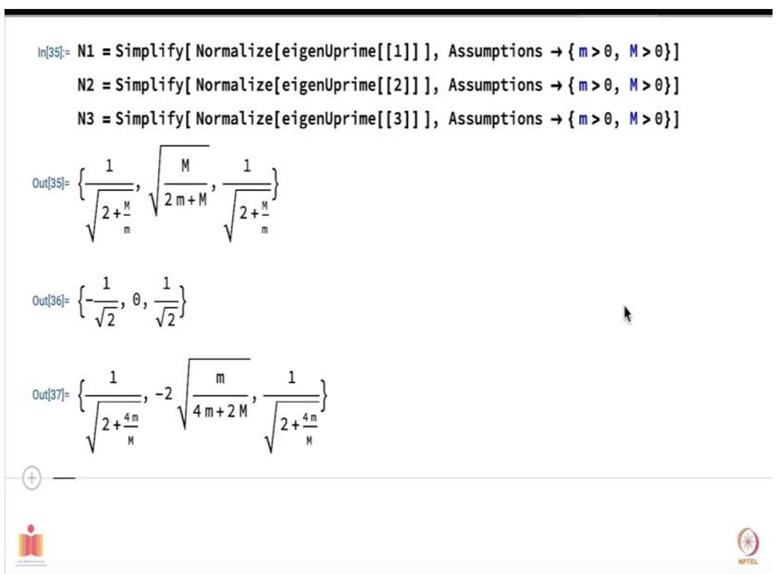


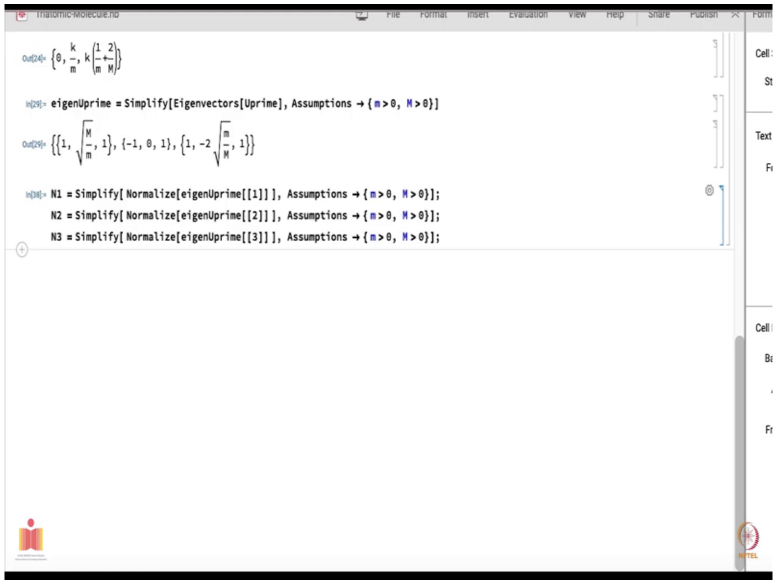
I will remove it and this is fine anyway. Now I say simplify and that simplify I can bring with these assumptions. So, I copy paste this thing. I could have defined this as a rule and could have used this.

But anyway, for now, I will just copy paste. Let us go, let us see. So, better than before. So, now it is, it is not worried so much about whether it is complex and it is taking the square root of complex number, it happily puts everything under the square root. That is nice. Let us see, now I should get all the three. So, three, all the three I want to normalize.

(Refer Slide Time: 19:12)

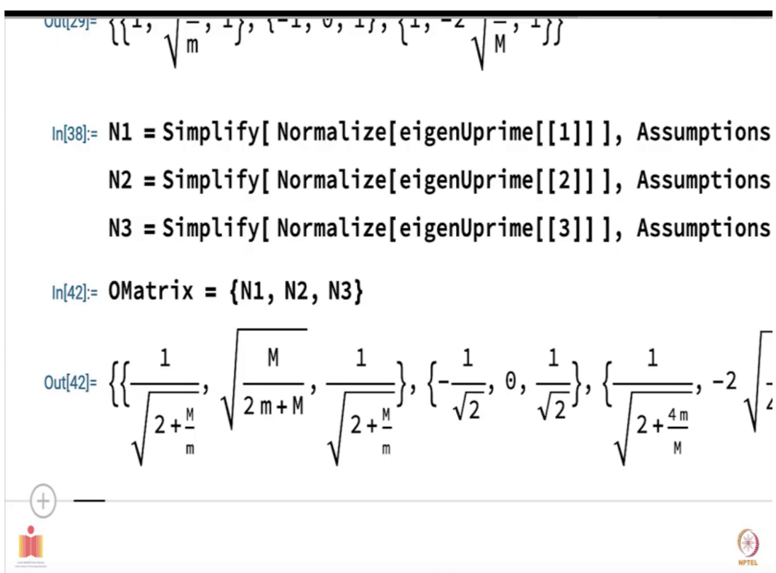


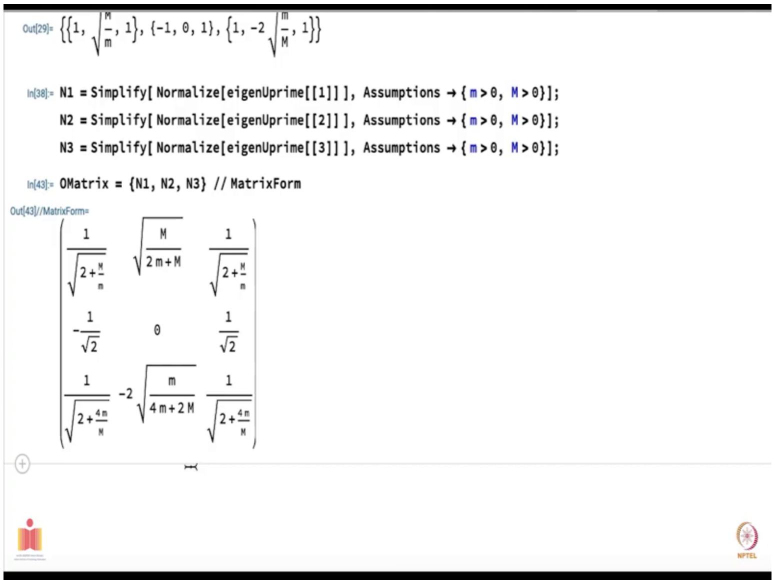




So, N1 is the normalized Eigen vector 1. I can just copy paste this. Of course, you can do it in a much better way, but I can just do simply here and N3, you see it shows in blue if it is it has not been executed. Now I will run all the three and let me put see that is correct. That is good, so we have this result now for these three Eigen vectors, which are normalized. Let me hide the results. I put it like this.

(Refer Slide Time: 20:01)



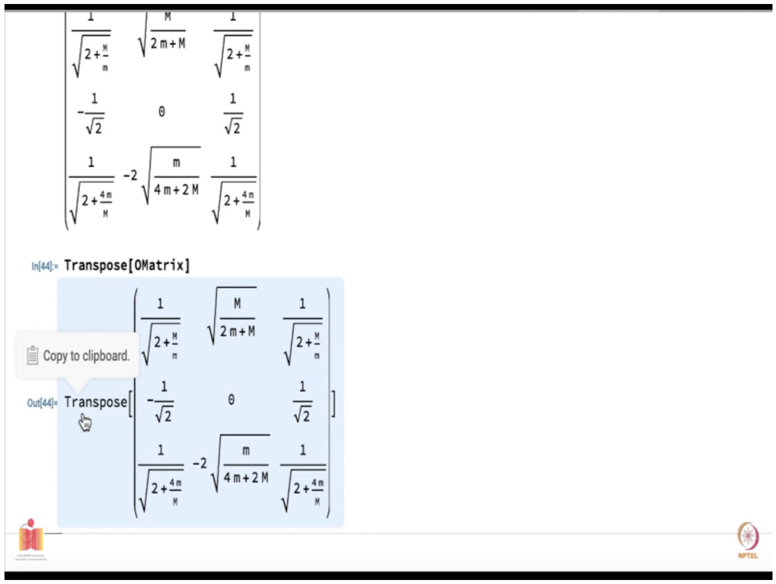


Now I construct an orthogonal matrix which I will call O, I hope it is not something built in. Let me see, just to test O equal to 1 does it work? No, It work. No O is protected. See, it says that symbol o is protected. So, I should not use this. I can define O metrics, then it is fine.

Now I can write it as. So my matrix O matrix, which is what was in the notes called O, was this matrix and the entries are the vectors N1, I think I am doing a mistake, I should not put all this N1, N2, N3. Let us see. I do get a matrix, let me put it in MatrixForm. In fact, let me just tell you that before giving that lecture, I had calculated everything in Mathematica to be sure of all the factors and all the competition and then only I had made my lecture instead of doing things by hand.

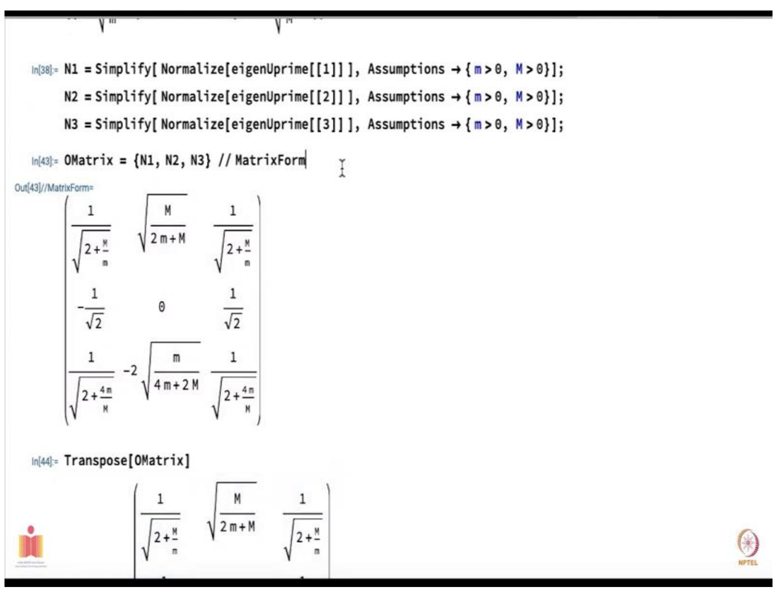
So, this will be useful for you also to practice. Now, this is the matrix. Let us check whether this is right. So, let us do some tests on this. Let me do something which most likely is not going to work and then we will fix it.

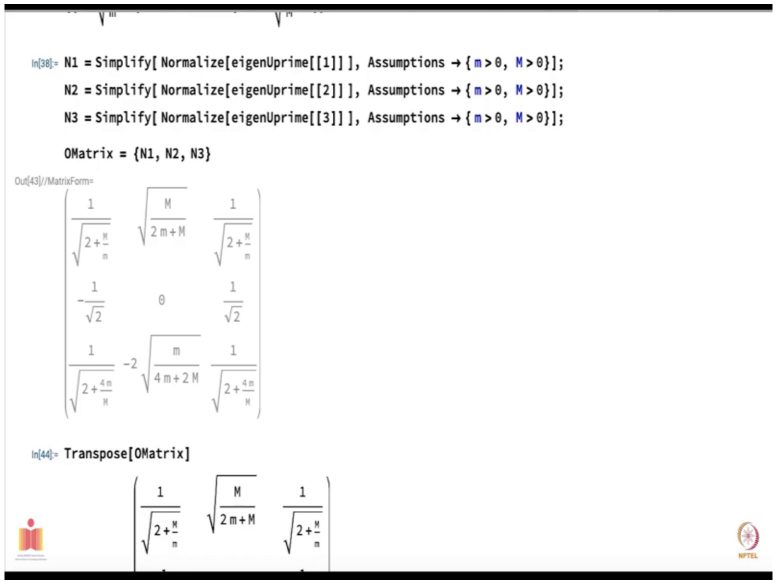
(Refer Slide Time: 21:39)



So, I say transpose of O matrix, transpose I want to find the transpose of this matrix. Let us see. Let that work, it should not. No, it worked that is (())(21:59). You see, it just left behind transpose.

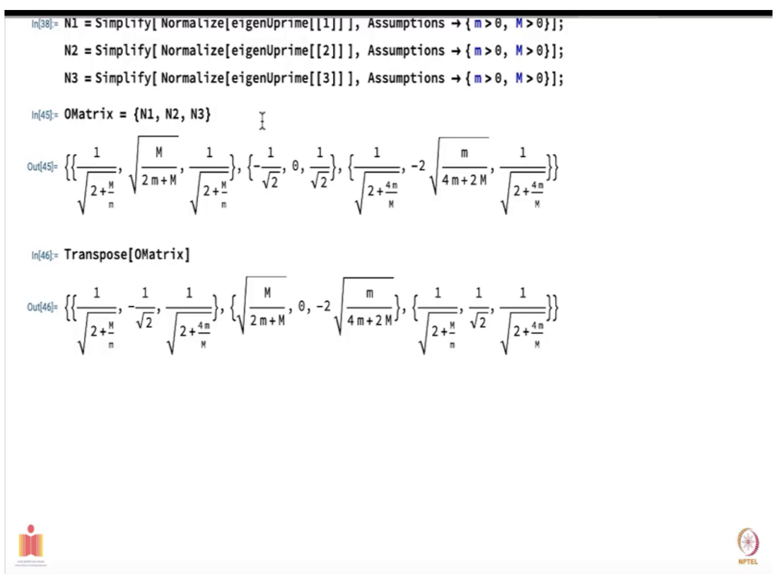
(Refer Slide Time: 22:07)

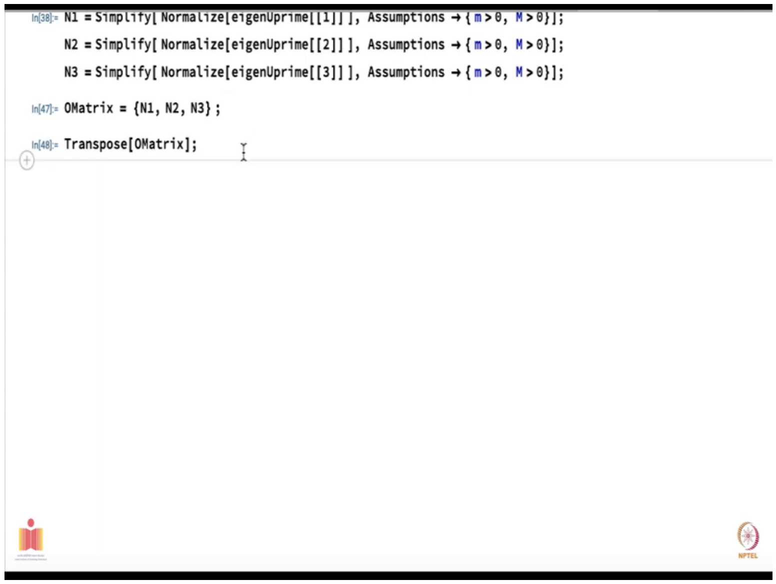




And the reason is this MatrixForm so I should remove it and do a Shift Enter. I cannot see it.

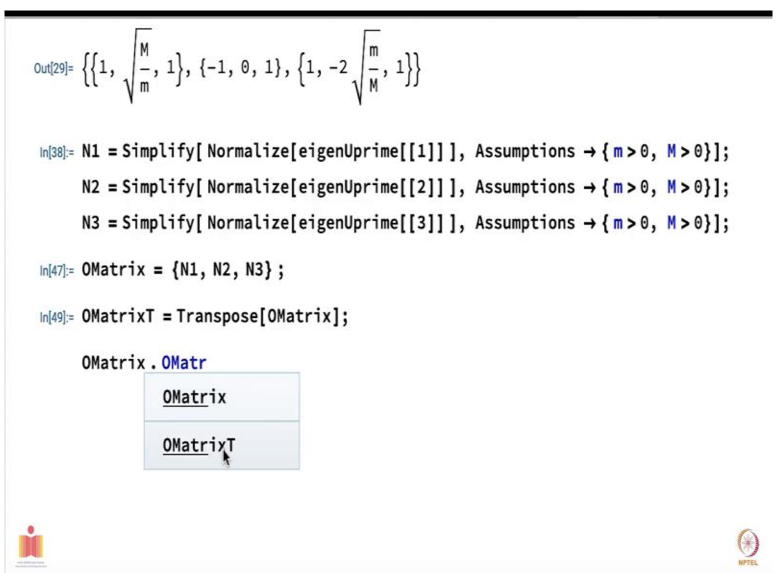
(Refer Slide Time: 22:15)

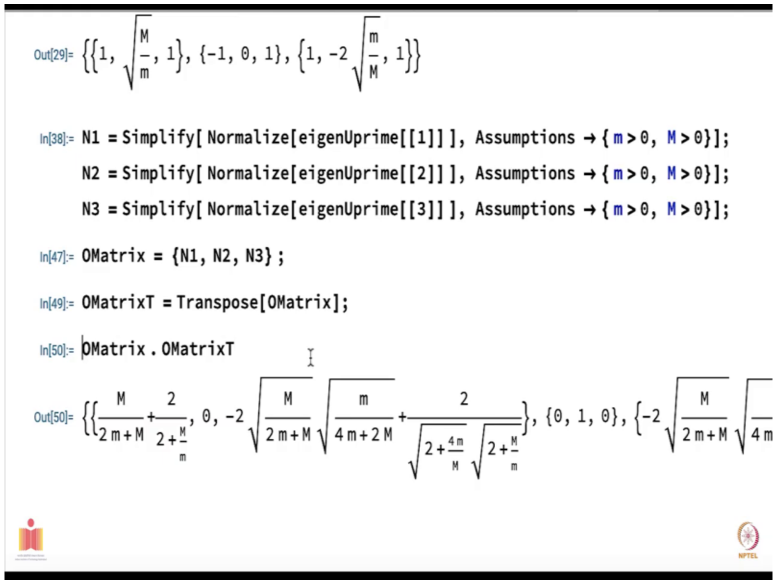


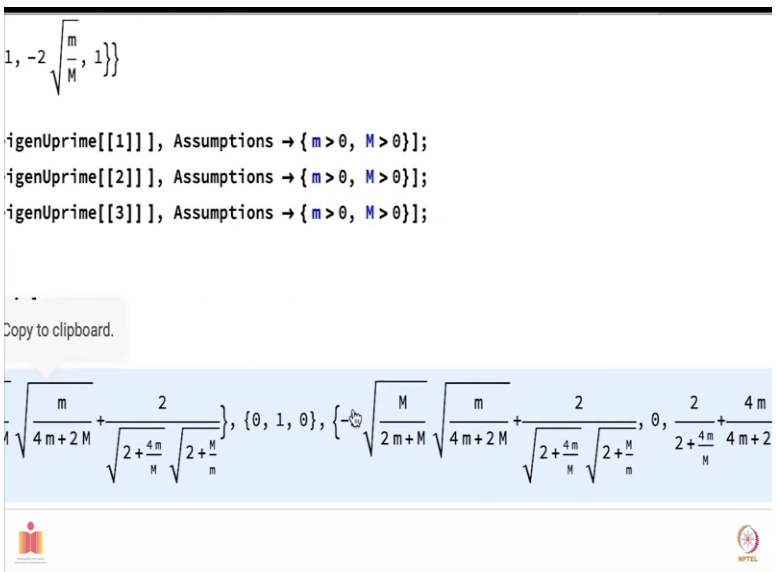


But I mean, I cannot see it nicely now. But the competition will happen. This is nice. So, I have got the O matrix, Orthogonal matrix and transpose matrix. Let me hide outputs, there is, I am not going to, do not really need to see them. Now, I should have given a name here. O matrix, that is what will be. Now I should run it again. Now, let us check whether the matrix which I have constructed it is really orthogonal.

(Refer Slide Time: 23:03)

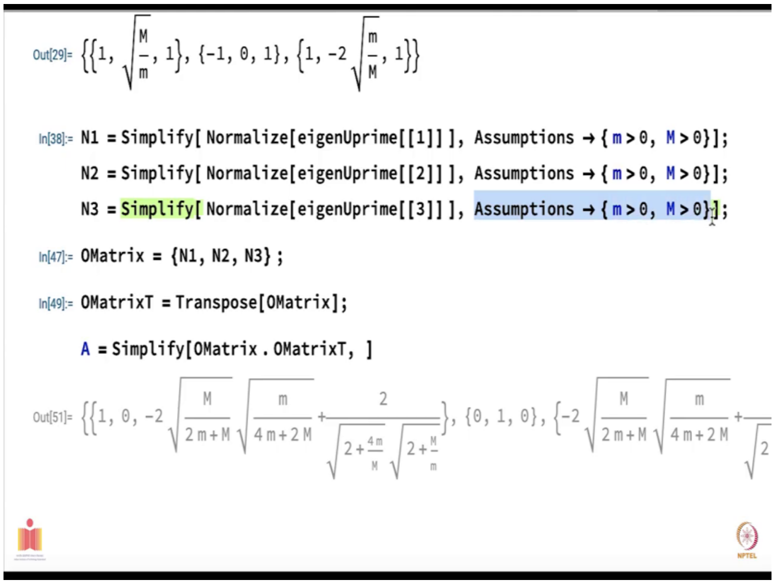


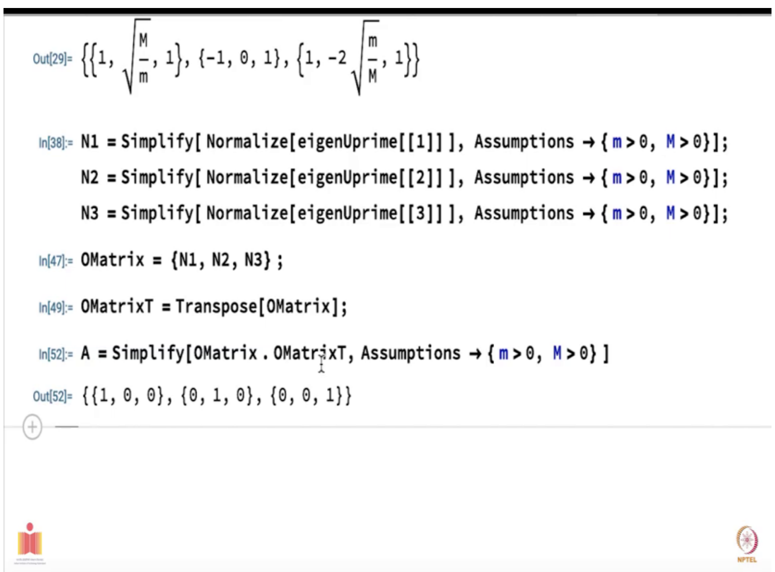




So, I should get O matrix dot O, see already shows this. I should get unity. It does not. Most likely. It is correct. Let us see. It does not look like everything is good, but I will. Let me call it, we will just simplify. Let us see whether it works. Let us see. So, I have first entry is 1.

(Refer Slide Time: 24:02)

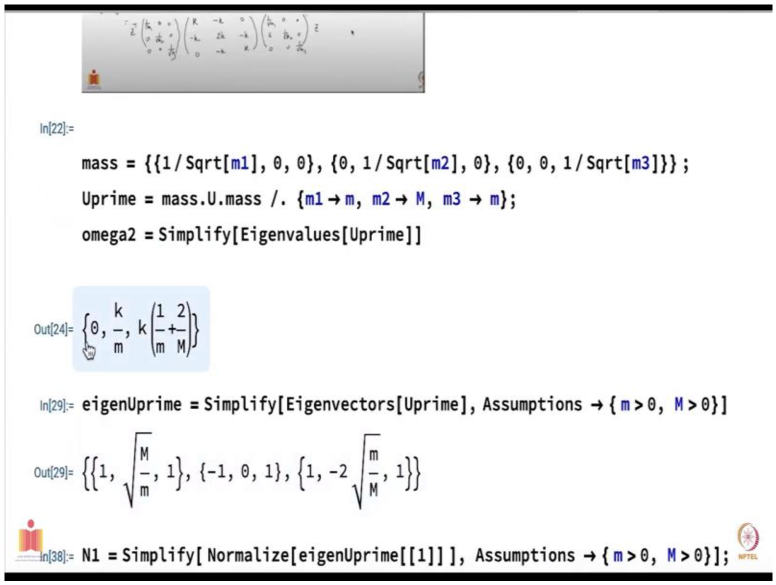


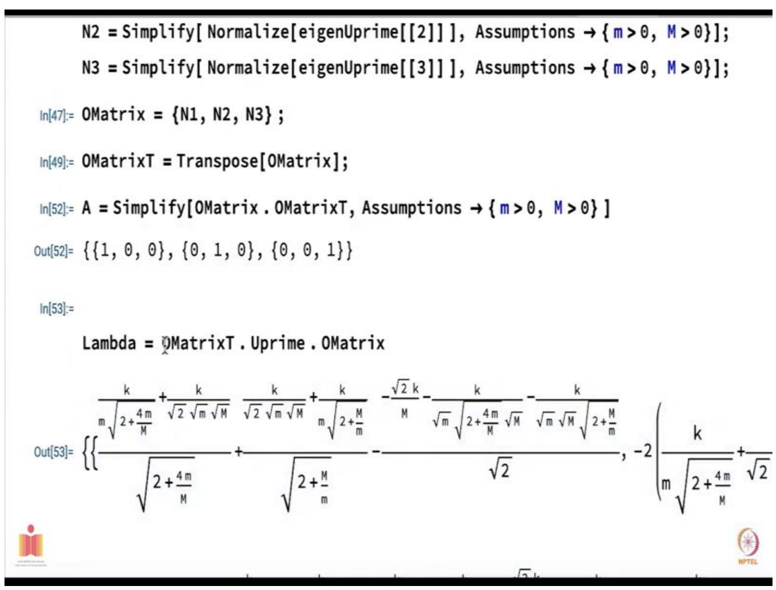


So, the let us call this matrix as not A that is a bad name to give, but A11 will be 1. So, this one is correct. The second is 0. We expect this thing to be 0. At least the signs are opposite that gives a confidence, M most likely these two terms will, the M goes up, (())(24:23) the M goes up to M plus for small m 2 capital M plus 4 small m this is correct. That is correct. This looks correct, actually, this is correct and we want to make it explicit term.

So, let us try this thing again. Hopefully it will work. Perfect you see this is a diagonal matrix now. So, we have checked that the matrix that I have constructed is orthogonal. If you wish, you can also multiply O transpose times O and check it will again come out to be identity. So, that is good. Now this matrix should diagonalize my U prime and if I remember correctly that is what we call lambda, so let us use lambda.

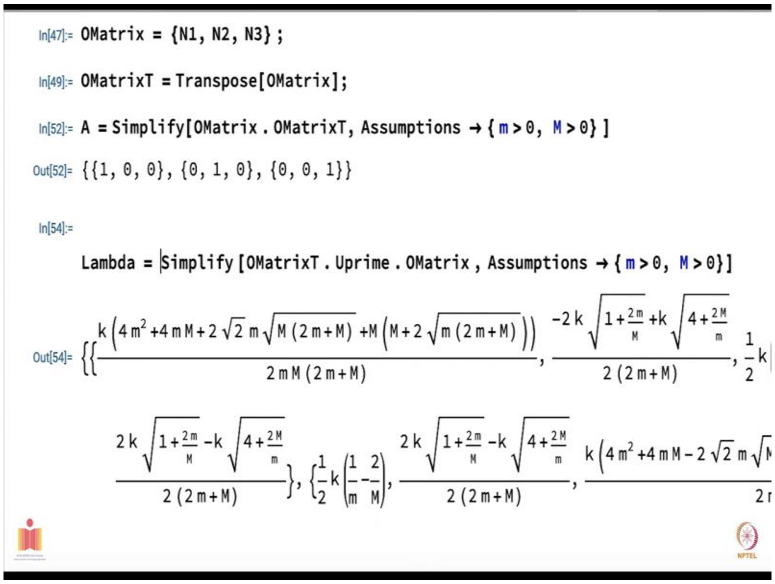
(Refer Slide Time: 25:30)





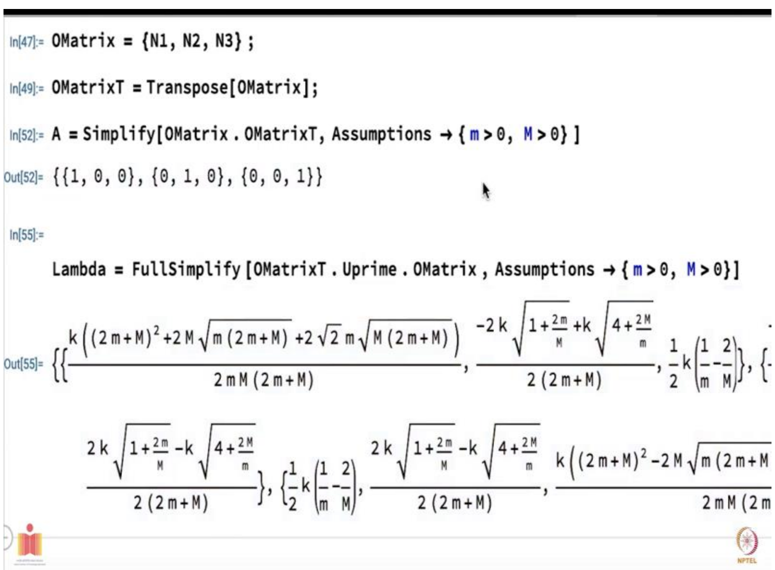
Let us create a new L A M B D A is equal to O matrix transpose O matrix t times U prime, I think I called it yes times O. If I have done everything correctly, I should get a diagonal matrix and the diagonal entries should be the Eigen omega squares which we have already calculated this thing here omega 2. This is what I should expect. Let us see whether that happens. It is not looking very nice.

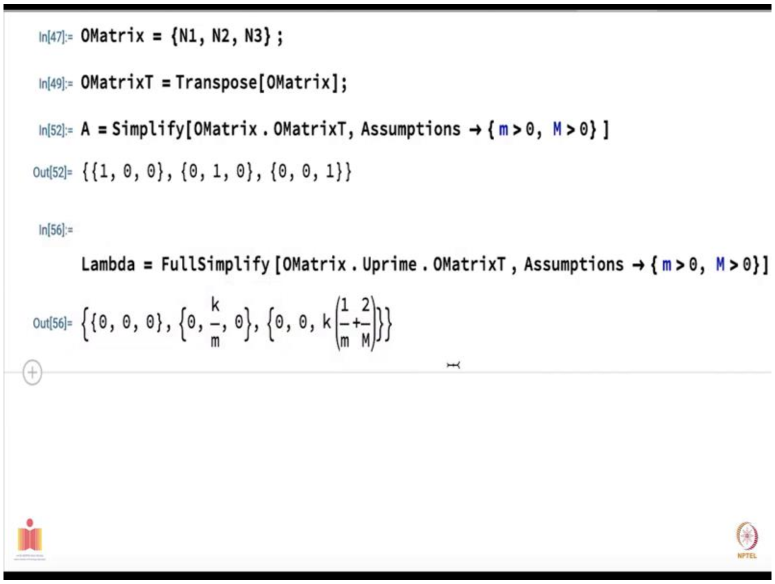
(Refer Slide Time: 26:25)



But now we have a lot of experience. I should just simplify it and again, as before with those assumptions, let us say yes. No. What happened? What it is worried about?

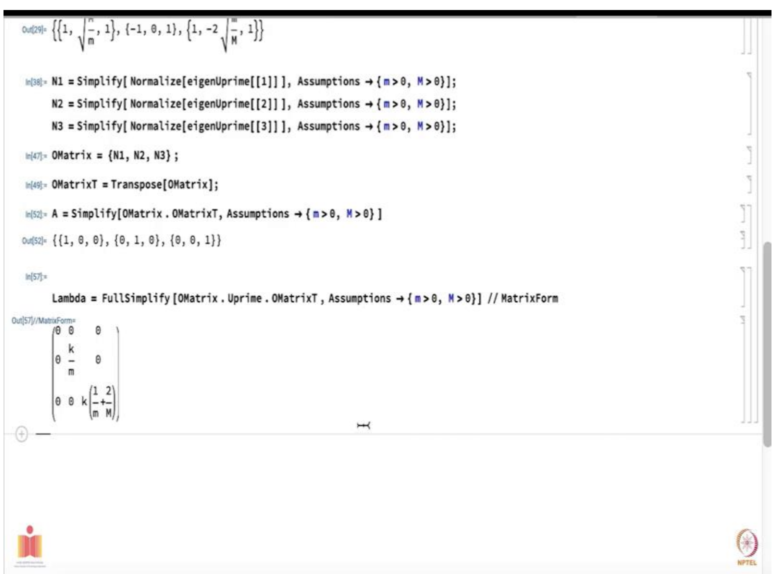
(Refer Slide Time: 26:56)

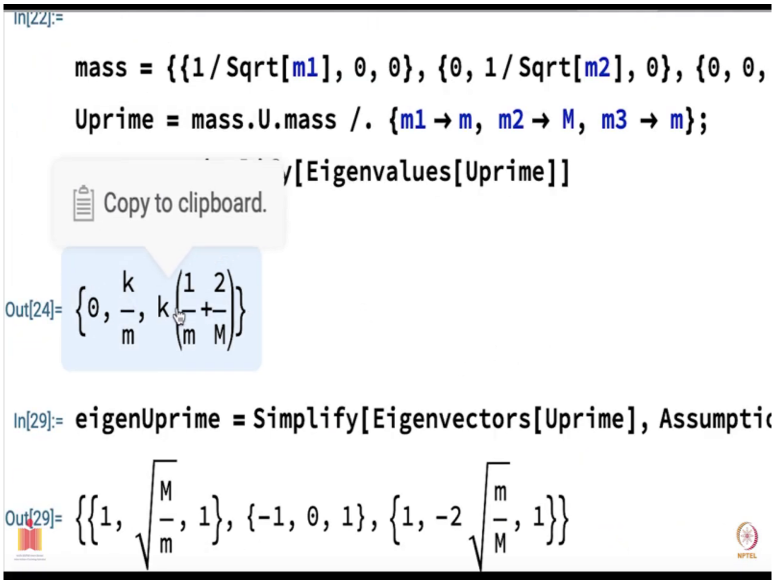




Maybe full simplfy. Not looking good. Actually, this is not at all looking good. Let us see what has gone wrong? Could it be this, let us check, that was a problem. So, I should have written O, U prime O transpose and I had done the opposite thing. But now with this, this is perfect.

(Refer Slide Time: 27:40)

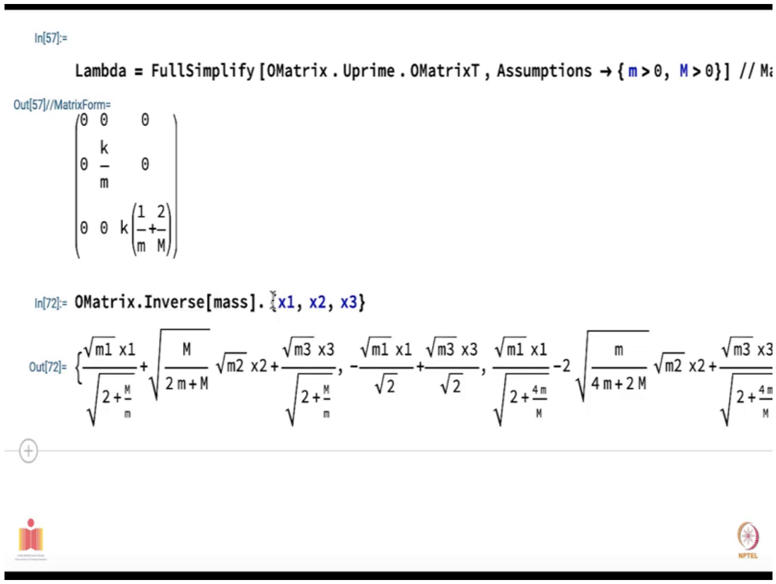


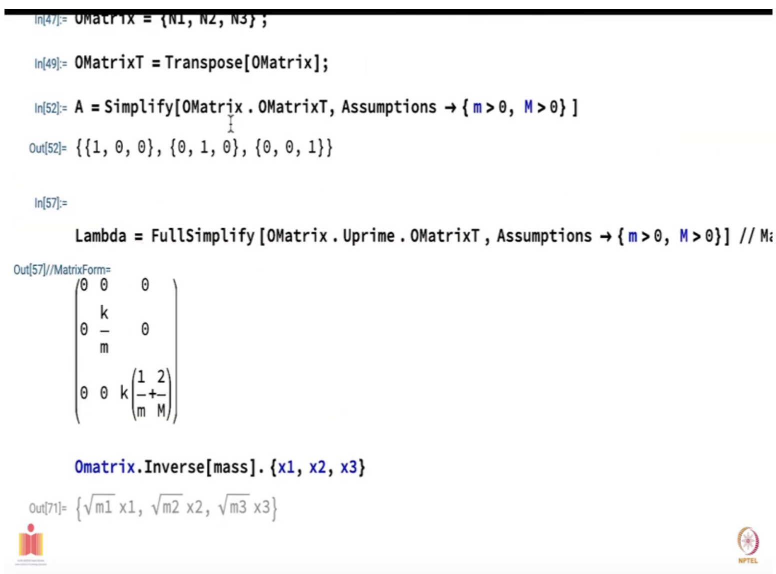


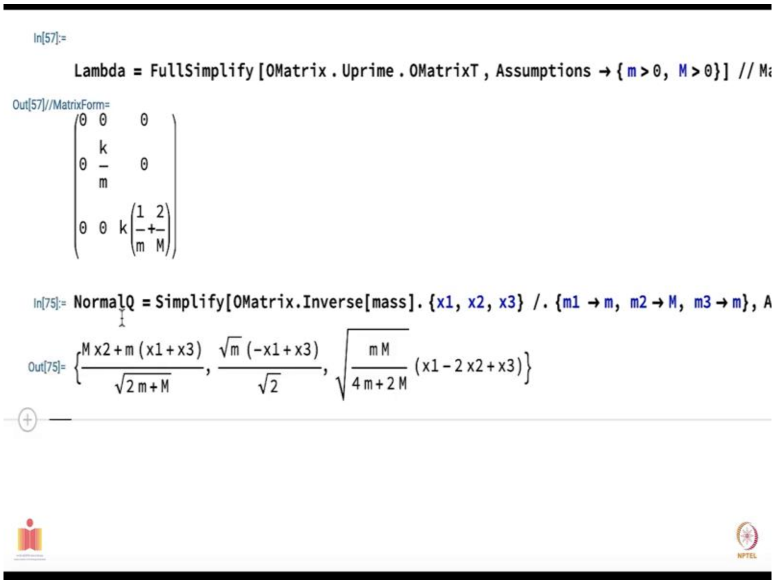
Let us see, I put in the matrix form just for a nice visualization by MatrixForm and yes the entries are diagonal and as you see, the diagonal entries are same as what you got before. Shall we go and check here. Yes, you see 0 k over m and k times 1 over m plus 2 over m and this is what you have here. So, everything has been correct till now. Which is nice, now that I have lambda, I am sure that q metrics which, O matrix, which I have calculated is correct.

Because my lambda has turned out to be diagonal and it shows the correct Eigen frequencies or square of the Eigen frequencies. Now, I want to find out the normal coordinates. So, let me do that.

(Refer Slide Time: 28:46)







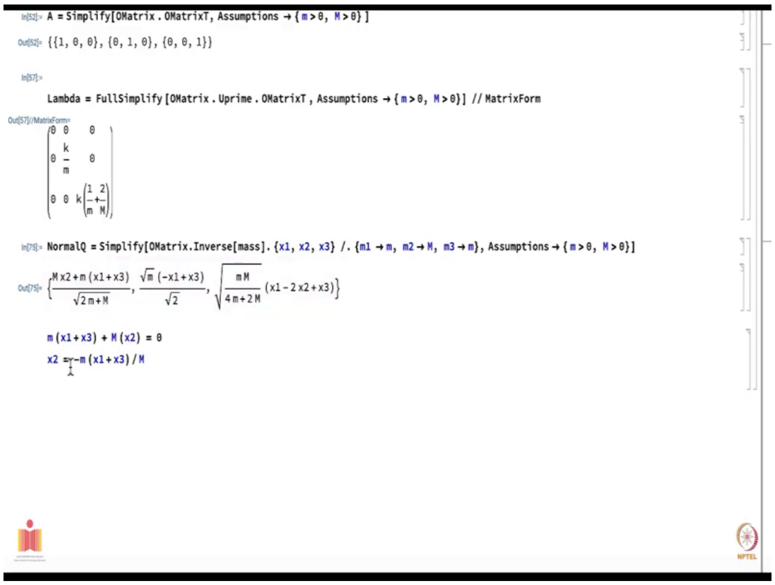
Normal coordinates, which I have denoted by Q and this is the following I can, I can do it in two steps, I can get rid of this first. So, I will need to multiply the inverse of that matrix, what I call mass. I need to take the inverse of it, so that it has diagonal entries that as square root of m1 square root of m2 and square root of m3.

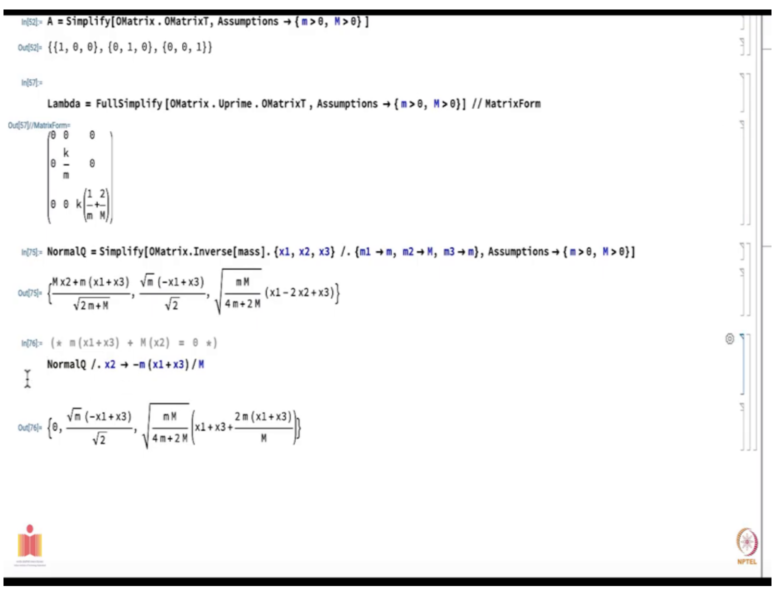
So, I can just take inverse of this matrix. Let us see what happens, because there is no e here. That is why, it could not do anything. Perfect. This is what we have to multiply and in this I should dot the vector column vector x1, x2, x3 correct and now if you recall what Q is Q for getting the Q I should multiply this entire thing with the Orthogonal matrix which I have written O matrix.

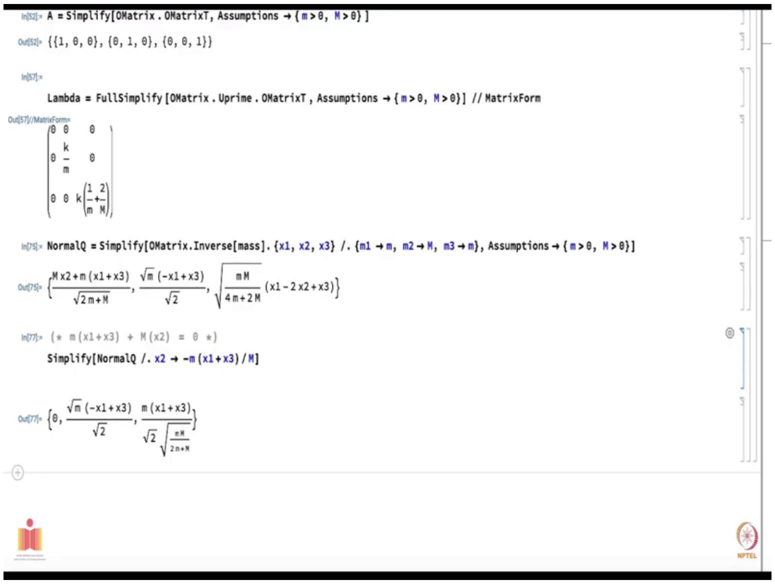
Orthogonal matrix O. O was M capital, actually it is better never type it fully, I can write O, it is not showing somehow. No problem, this is fine let us see what it is and one problem is because our m1, m2, m3 is appearing because in the mass when I had defined it was still m1 m2 m3 I had not made this as small m and capital M. So, let me do that now, m1 is m the middle one was capital M and m3 was again small m.

Let us see what happens now? Hopefully looks almost fine it is having the, let me go have to simplify with the assumptions and assumptions are that this is true. Let us see, let me check whether this is all okay, looks good, the third one is also. This is correct actually, this is correct. This is what I want to call as normal Q. So, I just run it again. So that is stored, that is the answer. But I can go one more step and try to impose the condition that center of mass is at rest. So, I want to or rather it set origin. So, I want to impose that.

(Refer Slide Time: 33:07)





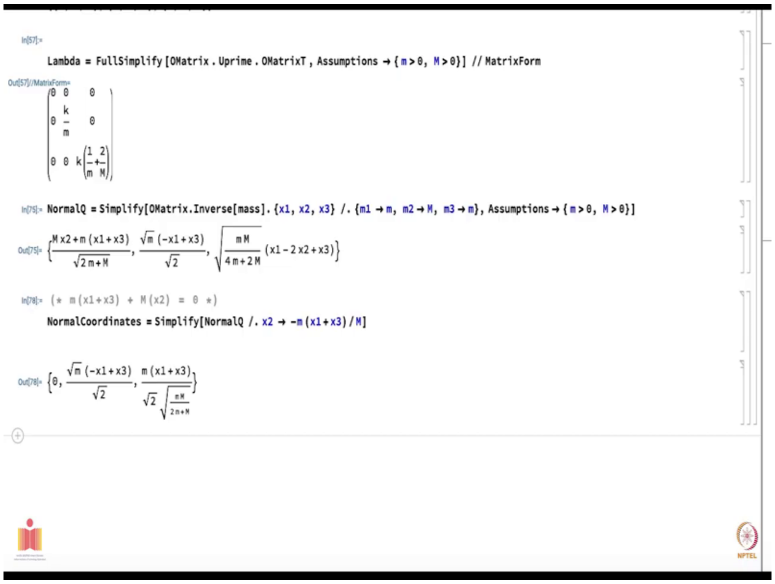


So, this is the thing which I want to impose m times x1 plus x3 plus capital M times x2 is equal to 0. This is, this is telling that I can put a bracket it will be easier to read. So, this is what I want to impose. So this look may change this to something here, there should not be a way to define this as a, somehow it is (())(34:00). Anyway, so I will leave this I wanted to make it as a comment, I do not know why I am not able to do it.

So, which means that this one what I really want to do is x2 is equal to minus m x1 plus x3 divided by capital M, is that correct? It minus m. This is what I want to impose. So, let me remove this line it is not necessary. There is a way I have to put a comment you can just put like this. You can put between these, you open a round bracket with a star and close it again with a star in a round bracket.

So, that becomes a comment. So this is what I want to do. So in the normal Q, I replace x2 by this. Let us see what happens that is looking of course good one of the coordinates have become, which corresponds to the center of mass. Let me simplify a bit, S I M P L I F Y. Let us see what happens, perfect. Now we can see very neatly. Let me give it a name normal coordinates.

(Refer Slide Time: 36:00)



So, here you see x3 minus x1 and you have x1 plus x3. So, these correspond to your asymmetric and symmetric modes, this is what we saw in our in the lecture. So, we have done the full competition here using Mathematica and that is mostly it and you can also try to do other problems, for example when you are looking at the inertia tensor for rigid bodies.

Again, if you are given a rigid body you can look at what it is inertia tensor would be and try to find the principal moments, find the principal axis using Mathematica or whatever other programming language you prefer. So, I would encourage you to use one of these languages or programming languages to do your competitions. I will stop this video here and see you in another video.