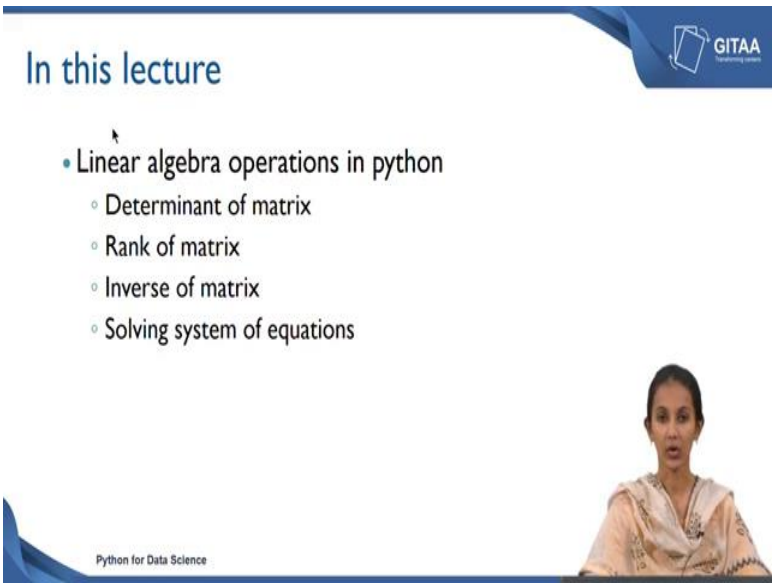
Python for Data Science
Prof. Ragunathan Rengasamy
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

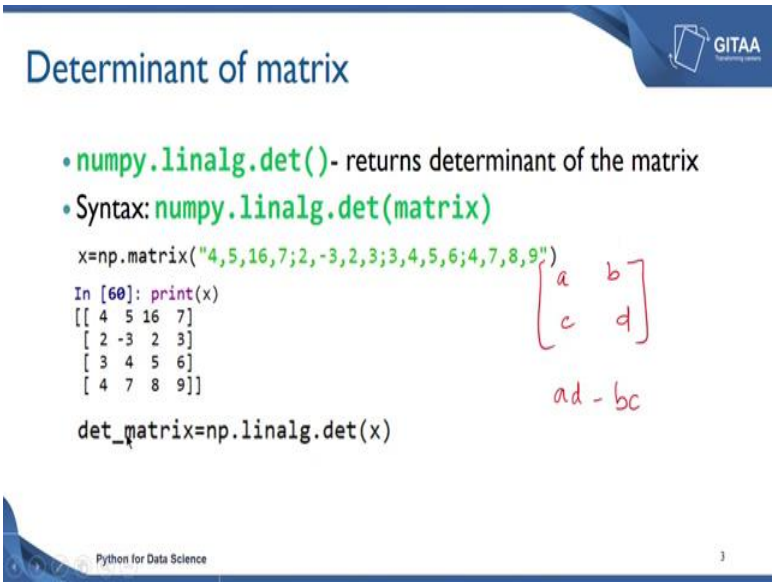
Lecture – 15
Linear algebra Part – 1

(Refer Slide Time: 00:17)



Welcome to the lecture. In this lecture we will see some of the Linear algebra operations in python. So, first one is determinant of matrix, how to calculate rank, how to calculate the inverse of a matrix and also how to solve the system of equations?

(Refer Slide Time: 00:37)



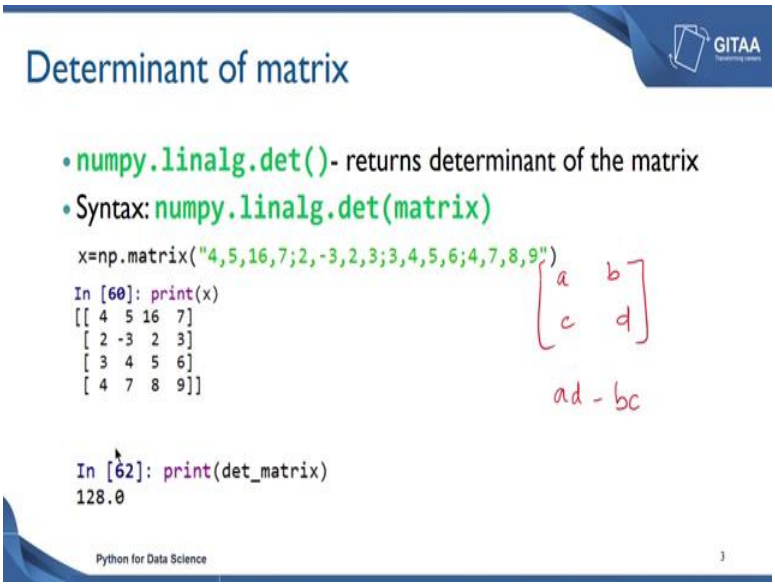
So, let us get started first we will look at the determinant of matrix to calculate a determinant of matrix a matrix it should be as square matrix. So, it can be a 2 cross 2 or it can be a 3 cross 3 or it can be a 4 cross 4 matrix. So, the determinant of matrix is very useful in calculating the inverse and also it is used in solving system of equations.

So, in python numpy.linalg.det it basically returns the determinant of the matrix. Let us say if I have a 2 cross 2 matrix a b c d, ad - bc. So, this is a way to calculate the determinant. So, if you have a 3 cross 3 also you can find the determinant you can also do it for 4 cross 4; as well the Syntax is numpy.linalg.det and inside the parenthesis we have to specify the matrix.

So, first we will create a matrix, I am going to create a 4 cross 4 matrix, which means 4 rows and 4 columns you can give values for the first row that is 4, 5, 16, 7 and then after 7 you have to separate it with semicolon. And again you have to specify the values for the second row and similarly for third row and fourth row here I am storing in variable call x matrix.

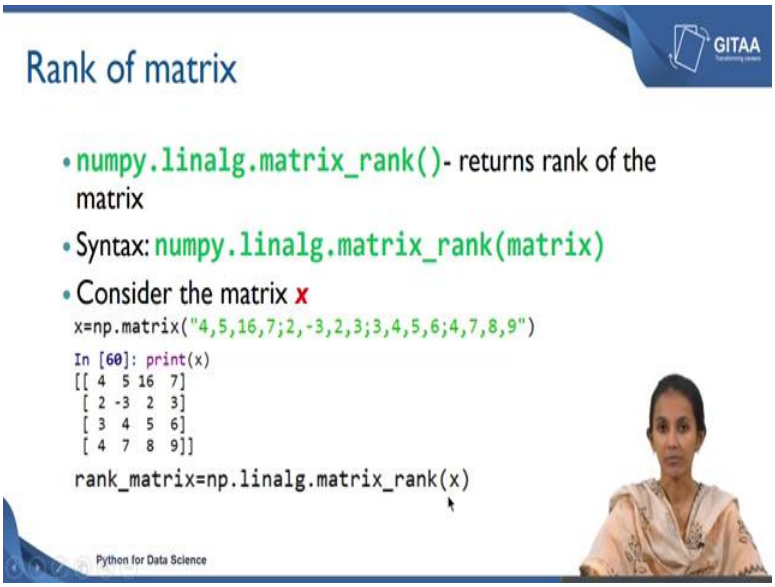
Now let us print the matrix x so, it has created a 4 cross 4 matrix which is a square matrix. Now let us calculate determinant for this matrix so numpy. So, before calling an numpy you need to import numpy. So, you have to import numpy as np; np.linalg.det(x) and I am storing in variable call det_matrix which is it determinant matrix.

(Refer Slide Time: 02:38)



Now, let us print the det_matrix the determinant value is 128.0, so this is a value for the 4 cross 4 matrix. Similarly you can create other matrix and you can calculate the determinant as well.

(Refer Slide Time: 02:51)

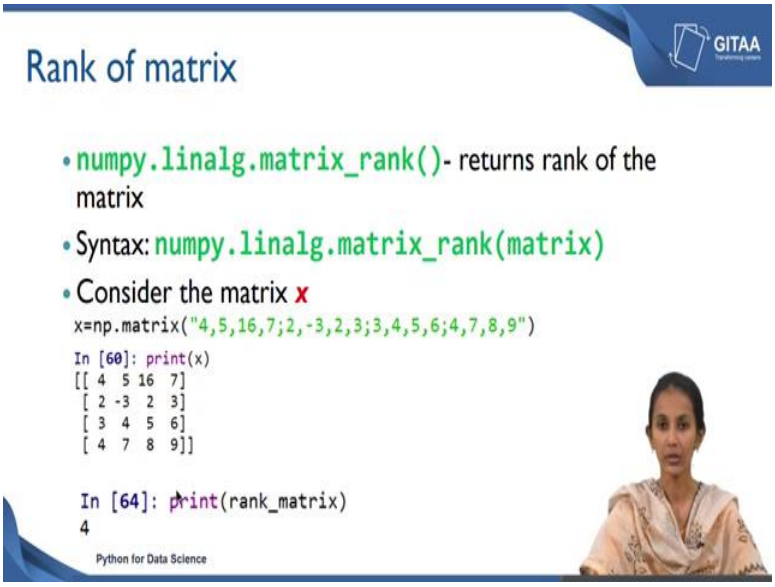


Next one is rank of the matrix. So, basically the rank is used to find the number of linearly independent rows or linearly independent columns. So, in python so again in the numpy package, we have the matrix_rank which basically performs the rank. The Syntax

is numpy.linalg.matrix_rank and inside the parenthesis again you have to specify the matrix name which you are going to create.

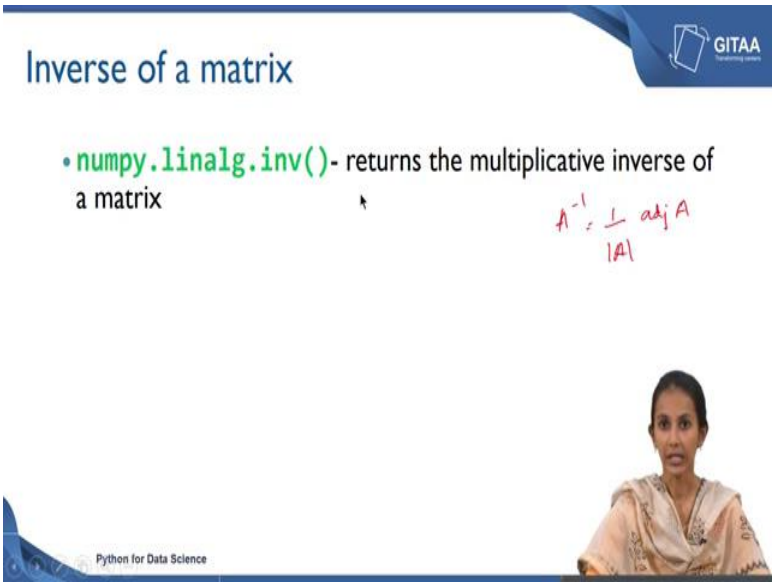
Now let us consider the matrix x same matrix we will find the rank for this matrix. So, it is a 4 cross 4 matrix right for this 4 cross 4 matrix calculate the rank. So, the command is np.linalg.matrix_rank. And inside the parenthesis you have to specify matrix name and I am storing it variable call rank_matrix. So, you can also store it in some other variable name as well.

(Refer Slide Time: 03:58)



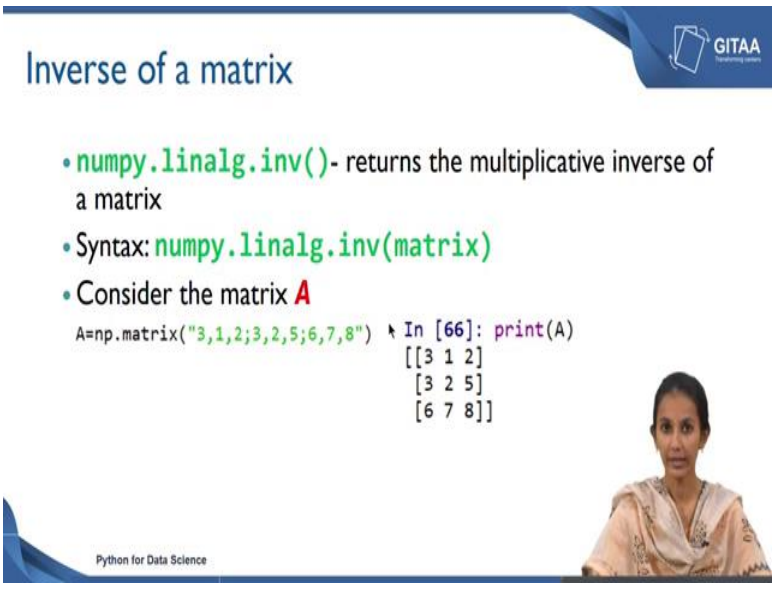
So, let us print the rank value so, it shows 4. So, which means it has 4 linearly independent rows.

(Refer Slide Time: 04:04)



Next we will look at the inverse of A matrix; inverse formula is A inverse is $A^{-1} = \frac{1}{\det(A)} * adj(A)$ so, this is the formula for the inverse of matrix. So, in python the command is.inverse it basically returns the multiplicative inverse of a matrix.

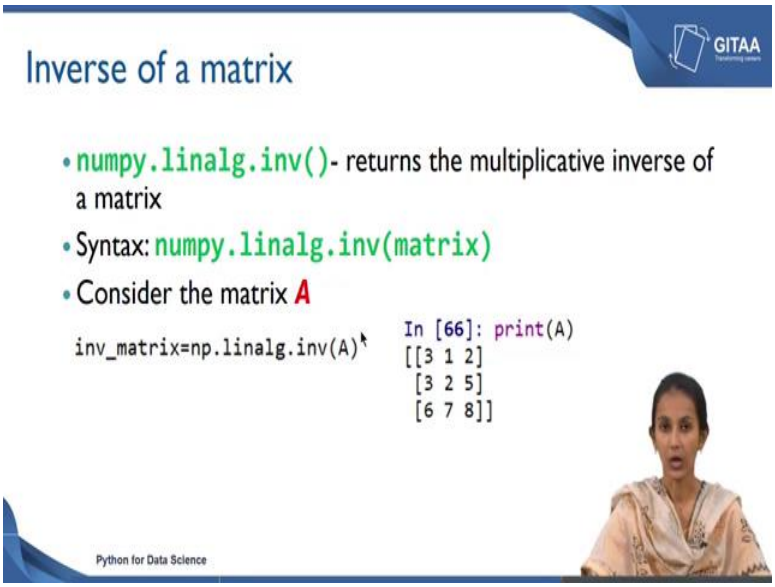
(Refer Slide Time: 04:32)



The Syntax is numpy.linalg.inverse that is inv and inside the parenthesis you have to specify the matrix name. I am creating a matrix 3 by 3 and then storing in a variable call in A. So, before calling np.matrix you have to import the numpy package import numpy as np and from the numpy package you can call the matrix.matrix. So, the values for the

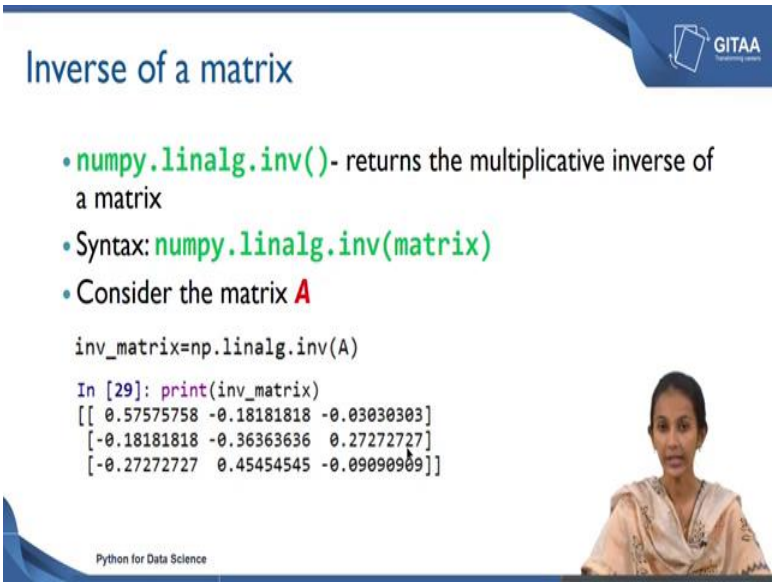
first row are 3, 1, 2; 3, 2, 5 for the second row basically for the third row. So, now, we have printed the values so, this is a 3 cross 3 matrix for this matrix we will calculate the inverse.

(Refer Slide Time: 05:15)



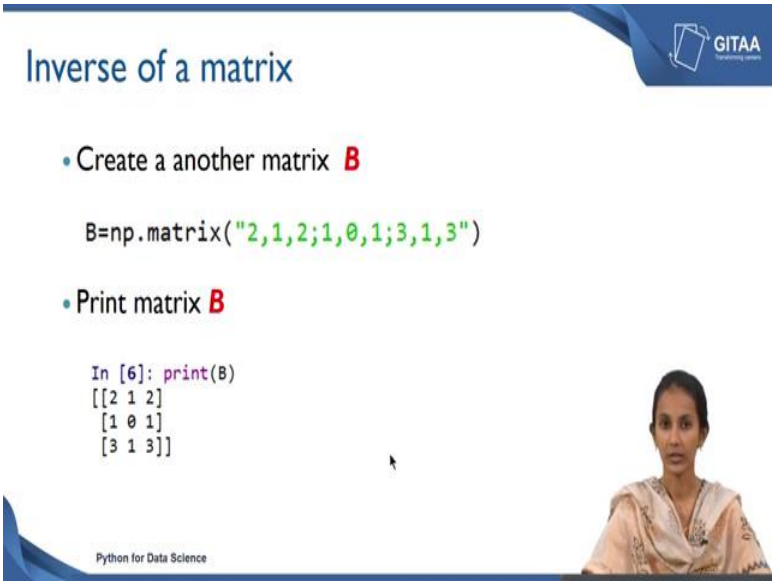
So, the command is np.linalg.in inv and inside the parenthesis I am specifying the matrix name which is A. And you can store it in the variable inv_matrix. So, now, let us print the inverse of the matrix.

(Refer Slide Time: 05:34)



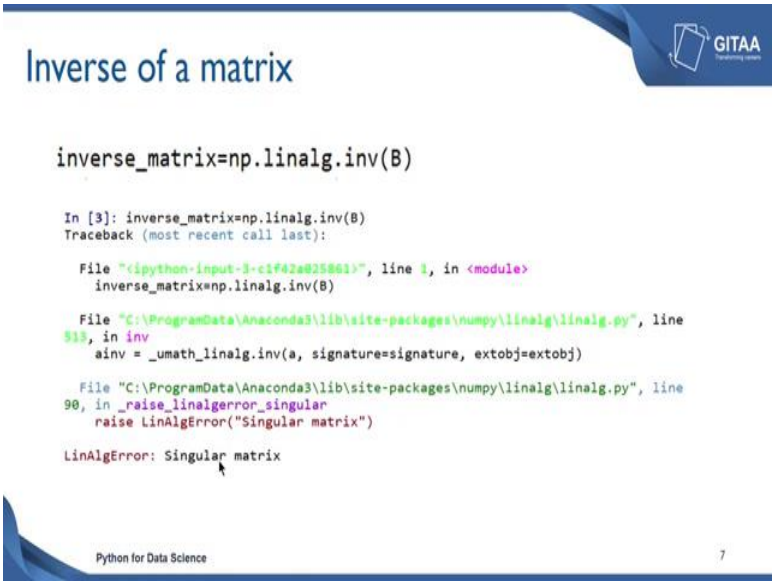
So, this is a value for the matrix which we have created. So, this as return A inverse. So, it as calculated 1 by determinant A into adjoint A and returns the output.

(Refer Slide Time: 05:47)



Now, let us take an another example for calculating the inverse. We will create another matrix B. So, this is again a 3 cross 3 matrix now let us print the matrix B. So, the values are 2 1 2; 1 0 1; 3 1 3. So, it is a 3 by 3 matrix.

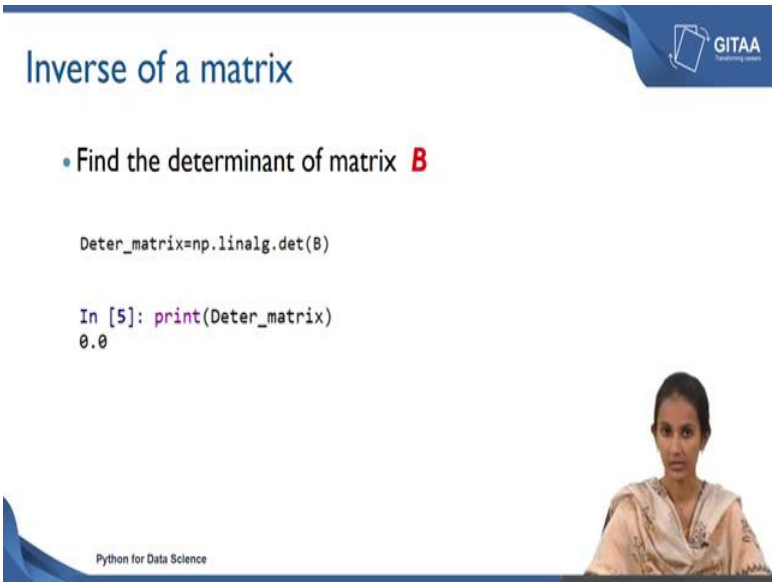
(Refer Slide Time: 06:12)



Now, let us calculate the inverse. So, the command is np.linalg.inv and inside the parenthesis here I am specifying the matrix B. Now, let us print the value of the inverse.

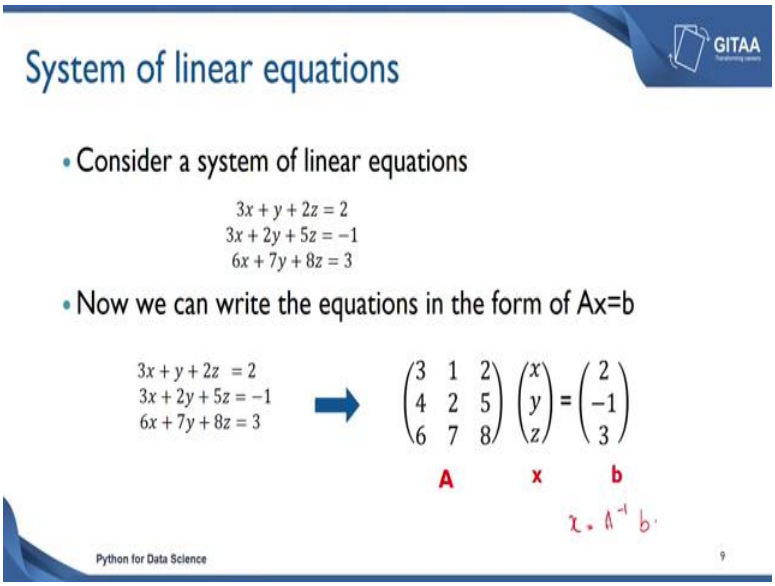
So, it has thrown a some error saying linalg error and it is singular matrix; its showing some error. So, we need to find what is that error right. So, the singular matrix in sense the determinant will be 0. So, if the determinant is 0 inverse is does not exist. So, we will not be able to find the inverse.

(Refer Slide Time: 06:48)



So, let us look at the determinant for this matrix. So, np.linalg.det which is a command for the determinant and inside the parenthesis I am specifying the matrix B. So, now, let us print the determinant value, the determinant value is 0 for in this case. So, that determinant is 0; so, the matrix will be a singular matrix condition to find the inverse is. So, determinant should not be equal to 0.

(Refer Slide Time: 07:14)

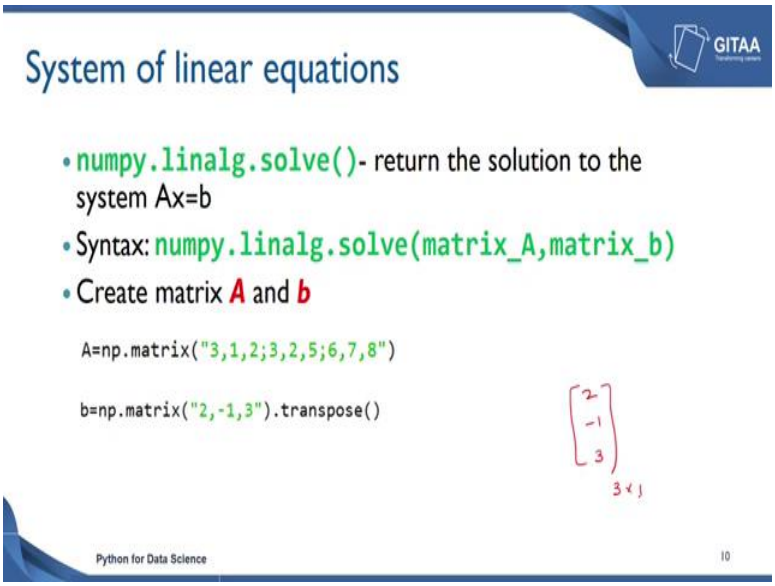


Now, let us look at the system of linear equation. So, if you have 2 or more equation linear equation then it is called as a system of linear equations. So, if you solve 2 or more equations, you can get the unique solution or there might be a no solutions or are there might be a infinitely many solutions. So, let us see how to do it in python. So, we will consider a system of linear equations. So, in our case taken 3 equations which is 3 x+y+2 z = 2 and the second equation is 3 x+2 y+5 z = - 1. Third equation is 6 x+7 y+8 z = 3.

So, these are the 3 equations, so we need to solve these 3 equations to find the value of the x,y,z. So, which are the basically the unknowns; so, we will write it in the form of Ax = B. So, these are the 3 equations now will write it in a x coefficients for x is 3 and respectively the coefficient for y is one and respectively the coefficients for z is 2 so, that has been written in the first row which is the values are 3 1 2. Similarly, you have to write the coefficients for the x,y,z for the second row and similarly for the third row.

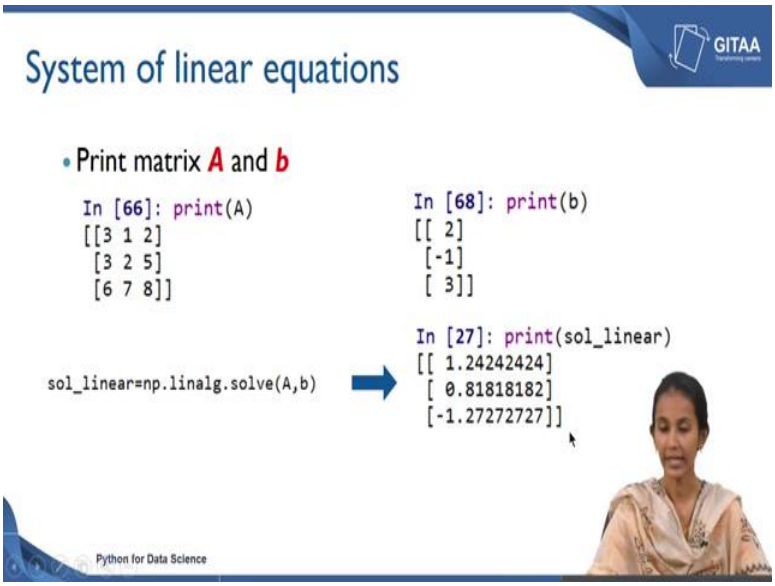
So, A are the coefficient values and x,y,z are the unknowns and b are the constant values which is 2 - one and three. So, now, we have written the equations in the format of ax = b now we will solve. So, we will keep the x on this side and will bring the A 2 on the other side so that becomes A inverse of b. So, we need to solve A inverse b; so, we can find the A inverse and then you can multiply with B or else there is a direct command which does the a inverse into B.

(Refer Slide Time: 09:18)



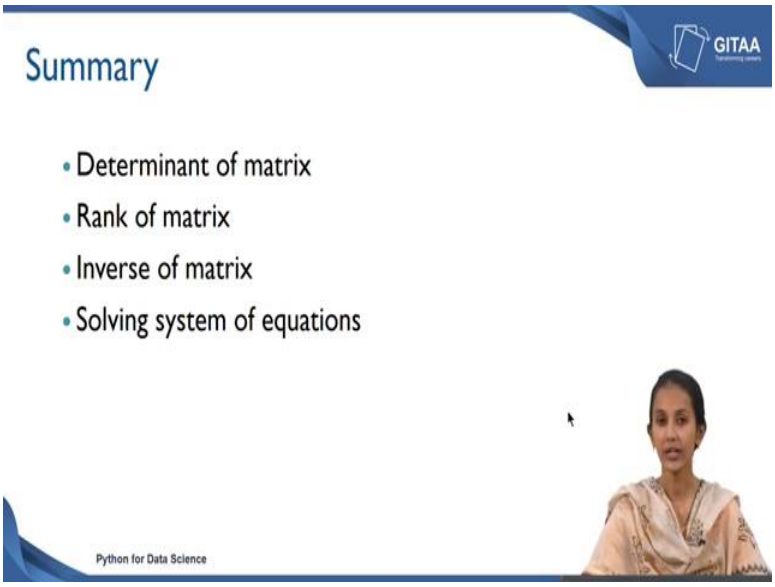
So, in python numpy.linalg.solve it basically return the solutions for the system of equations on the format of $Ax = b$. The Syntax is numpy.linalg.solve it basically solves the 2 matrices which is matrix, we have to specify the matrix A and the matrix b. Now, let us create a matrix A and b. So, A is 3 cross 3 matrix, so we will supply the values. So, after the end of the each row you have to specify the semicolon. Similarly you have to create a matrix b; in our case so, we had values 2, - 1 and 3 which is along the columns, which means we had 3 rows and 1 column so, that is why we have used the transpose to create a matrix b.

(Refer Slide Time: 10:18)



So, now let us we will print the matrix A and b which we were created. So, the A matrix will be a 3 cross 3 matrix and the b matrix will be a 3 cross 1 matrix. So, the command is np.linalg.solve and inside the parenthesis we have to specify the matrix which we have created earlier. So, we have created a A matrix and the b matrix; and we can store it in a variable. Now, let us print the values. So, when you print sol_linear. So, it has printed these 3 values. So, these are the values for the x y and z.

(Refer Slide Time: 11:00)



Let us summarize. So, first we saw how to calculate the determinant of matrix and we also saw how to calculate ranks. So, rank it basically gives the number of linearly independent rows or columns. And we also saw how to calculate the inverse. So, $A^{-1} = \frac{1}{\det(A)} * adj(A)$ and then we also saw of how to solve the system of equations.

So, system of equation sense $Ax = B$.So, you have to keep the x on one side. And you have to take the A on the other side. So, it becomes x = A inverse b and then you can get the values for x y and z.

Thank you.