

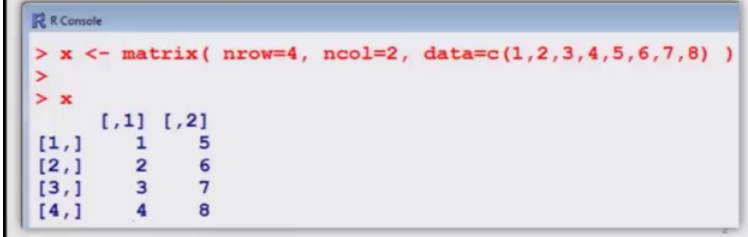
Introduction to R Software
Prof. Shalabh
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Lecture – 07
Matrix Operations

Welcome to the next lecture on Introduction to R Software. In the earlier lectures, we were trying to discuss about the matrix operations and we had discussed simple operations like addition, subtraction, multiplication and so on. In this lecture we will continue with some more basic operation, and I would like to show you that what are the different features available in R with respect to the matrix operation world? There are so many operations, but here my objective is to give you a glimpse of some of the important commands and to show you how to operate them on the platform of R. So, let us try to start now.

(Refer Slide Time: 01:02)

```
In R, a 4 × 2-matrix X can be created with a following command:  
> x <- matrix( nrow=4, ncol=2,  
               data=c(1,2,3,4,5,6,7,8) )  
> x  
      [,1] [,2]  
[1,]    1    5  
[2,]    2    6  
[3,]    3    7  
[4,]    4    8
```



The screenshot shows the R Console window with the following text: `> x <- matrix(nrow=4, ncol=2, data=c(1,2,3,4,5,6,7,8))`, `>`, `> x`, and the matrix output: `[,1] [,2]`, `[1,] 1 5`, `[2,] 2 6`, `[3,] 3 7`, `[4,] 4 8`.

So, right you may recall that in the earlier lecture, I had created a matrix x which was a 4 by 2 matrix in which I had taken the 4 rows and 2 columns, and I had considered the data 1, 2, 3, 4, 5, 6, 7, 8 and this matrix was coming out to be like this.

So, I will try to continue with the same matrix that we had constructed in the last lecture. And then we try to see that, what are the different aspects of this matrix that can be

revealed by R commands? For example, there is a command `dim`, which means dimension.

(Refer Slide Time: 01:35)

Properties of a Matrix

We can get specific *properties* of a matrix:

```
> dim(x) # tells the
[1] 4 2    dimension of matrix
```

```
> nrow(x) # tells
[1] 4      the number of rows
```

```
> ncol(x) # tells
[1] 2      the number of columns
```

R Console

```
> dim(x)
[1] 4 2
>
> nrow(x)
[1] 4
>
> ncol(x)
[1] 2
```

3

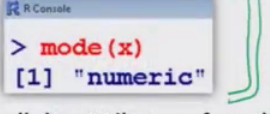
This dimension command helps us in getting the dimension of a matrix. So, the syntax is like this that first you write `dim`, `dim` and inside the bracket you have to write the name of the variable that is, containing the matrix. For example, in this case we have seen that our matrix here is for example, is of order 4 by 2. So, as soon as I say here dimension of `x` it gives me 4 2, this means 4 by 2 there are 4 rows and 2 columns. Now, in case if I want to know, what is the number of rows in a matrix then the command is `n row`. That is the number of rows and syntax is `n r o w` and inside the bracket you have to write down the variable name that is containing the matrix.

So, here we have seen that there are 4 rows. So, as soon as you try to do here `n row x` and will give you an answer 4. Similarly the number of columns of a matrix can be determined by the command `n col`; that means, the number of columns and inside the bracket you have to give the variable that is containing the matrix. So for example, here in this case that there are two number of columns. So, as soon as used right `n col` and inside the bracket `x` it will give you an answer 2, which is the number of columns. And here you can see that is the same thing I have done here that I have given the screenshot. So, if you want to try yourself you can also try these commands on your computer.

(Refer Slide Time: 03:21)

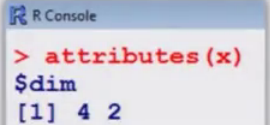
Properties of a Matrix

```
> mode(x) # Informs the type or storage mode of an object, e.g., numerical, logical etc.  
[1] "numeric"
```



attributes provides all the attributes of an object

```
> attributes(x) #Informs the dimension of matrix  
$dim [1] 4 2
```



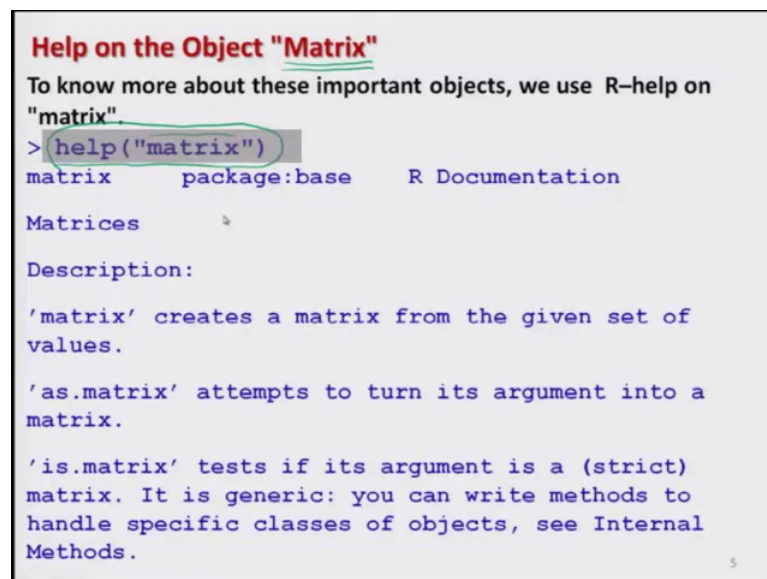
Similarly, there is another command that we had discussed earlier also, but just for the sake of quick revision I was try to show you here this is mode of x. Mode of x means this provides us the information that, what type of variable it is and what type of a storage mode of an object is contained in that variables name. For example, this can be a numerical value this can be a logical value and so on.

So, for example, here we have seen that in our matrix x, this is a 4 by 2 matrix containing some numbers from 1 to 8. So, this is a numerical value. So, as a as soon as I say here mode of x mode and inside the bracket I have to write the variable name that is containing the matrix. Then it gives me an answer numeric; that means, all the values contained in the matrix are numeric, they have a numeric character. And similarly attributes is another command, this provides some other type of information about the attributes of an object, which are controlled by that matrix.

For example, in our case x is a 4 by 2 matrix which is of numerical values. So, as soon as I say here attributes a double tributes and inside the bracket I have to give the name of the variable that is containing the matrix and as soon as I do so, it is giving me this type of output and it is trying to tell us, that the dimension of the matrix x are 4 by 2. Well, here I am trying to give you an example where I am getting only the information about the dimension, but similarly other type of information about the properties of a matrix can also be obtained through the attribute command.

And here also you have seen that, I have given here the screenshot of mode of x and screenshot of your attribute of x well these are very simple commands. So, instead of I am doing it here on the screen and showing it to you, I request you that you please try to type these commands on the R console and try to have an experience yourself. You will get exactly the same outcome, which I have shown here on the screen as a screenshot. Now there is another thing.

(Refer Slide Time: 05:50)

A screenshot of the R help page for the 'matrix' object. The title is 'Help on the Object "Matrix"'. Below the title, it says 'To know more about these important objects, we use R-help on "matrix"'. The command '> help("matrix")' is shown in a terminal window. The output shows 'matrix' from the 'package:base' R Documentation. The section 'Matrices' is followed by a 'Description:' section. The description lists three functions: 'matrix' creates a matrix from the given set of values; 'as.matrix' attempts to turn its argument into a matrix; and 'is.matrix' tests if its argument is a (strict) matrix. It is generic: you can write methods to handle specific classes of objects, see Internal Methods. A small number '5' is visible in the bottom right corner of the screenshot.

```
Help on the Object "Matrix"
To know more about these important objects, we use R-help on
"matrix".
> help("matrix")
matrix      package:base      R Documentation

Matrices

Description:

'matrix' creates a matrix from the given set of
values.

'as.matrix' attempts to turn its argument into a
matrix.

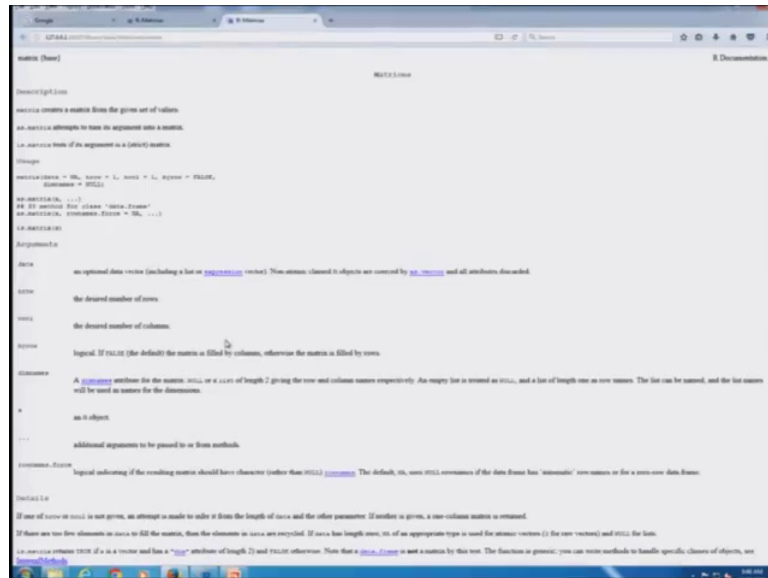
'is.matrix' tests if its argument is a (strict)
matrix. It is generic: you can write methods to
handle specific classes of objects, see Internal
Methods.
```

Now, we come to a more general aspect. Up to now, I have used the command matrix and I have taken different types of examples, to show you that, how our matrix can be created? How the data can be entered into a matrix in R? And how we can control the number of rows, number of columns and data? And whether the data has to be entered row wise or say column wise. I have taken all these aspects through simple examples. But, definitely we would like to know, what is the general command to enter the data into a matrix? And what are the different options available. Well, this is a very exhaustive topic and it is not possible for me to describe here all the commands.

But, I will definitely try to show you, how you can learn yourself and how you can explore that what are the different other aspects contained in the command matrix. So, in order to get some help on the object, matrix here we have to use the same rule that we had discussed couple of lectures back, that how to obtain the help on a particular object. Suppose, I want to know about the matrix so, I have to type here the command help and

inside the bracket within the double quotes I try to write down matrix matrix. And let us try to see what happens. So, I try to type copy this command and I try to type it over R you see, as soon as I do so the control goes over the internet and it gives me all information about the matrix. Well, it may not really be clear to you from the screen.

(Refer Slide Time: 07:37)



But, I have taken a screenshot of all this information and I will show you, but what I wanted to show you that you see all such detailed information is available. And now what I have done? I simply have taken the screenshot of this thing so, that I can explain you more easily over here.

So, this is here the mean I have simply copied and taken a screenshot. So, this is the first screenshot where it is trying to give different information that this is a package that is contained in the base under the R documentation. And yeah means what is matrix? Matrix is trying to create metrics from the given set of values. And you can see here, there are so many information and it is not only on this screen, but it is continuing in several other screenshots.

(Refer Slide Time: 08:37)

Then we get an overview on how a matrix can be created and what parameters are available:

Usage:

```
matrix(data [= NA, nrow = 1, ncol = 1, byrow = FALSE, TRUE, dimnames = NULL),  
as.matrix(x)  
is.matrix(x)
```

Arguments:

- data: an optional data vector.
- nrow: the desired number of rows
- ncol: the desired number of columns
- byrow: logical. If 'FALSE' (the default) the matrix is filled by columns, otherwise the matrix is filled by rows.

dimnames: A 'dimnames' attribute for the matrix: a 'list' of length 2.

x: an R object.

6

For example, here you can see me in the continues, the part is: this is the usage command: it is trying to give you here something like this. That it is trying to give you here the data, number of rows, number of columns, and then you have to write it whether the values have to be arranged by row, whether this is true or false.

So, if you want to arrange the data by rows, then you simply have to use here true; else if you want to arrange your data column wise, then you simply have to make it here by row equal to false and that is the default command also. And here it is trying to give you all sorts of thing that, what is the meaning of this here data? What is the meaning of here nrow? What is the meaning of here n col? And what is the meaning of here by row? Right. And similarly if you try to move forward yeah; it gives here more details about all the definition which are contained in the screen tags right.

(Refer Slide Time: 09:42)

Then, the meaning of each parameter is explained:

Details:

If either of 'nrow' or 'ncol' is not given, an attempt is made to infer it from the length of 'data' and the other parameter.

If there are too few elements in 'data' to fill the array, then the elements in 'data' are recycled. If 'data' has length zero, 'NA' of an appropriate type is used for atomic vectors and 'NULL' for lists.

'is.matrix' returns 'TRUE' if 'x' is a matrix (i.e., it is not a 'data.frame' and has a 'dim' attribute of length 2) and 'FALSE' otherwise.

'as.matrix' is a generic function. The method for data frames will convert any non-numeric/complex column into a character vector using 'format' and so return a character matrix, except that all-logical data frames will be coerced to a logical matrix.

7

(Refer Slide Time: 09:44)

Finally, references and cross-references are displayed...

References:

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also:

'data.matrix', which attempts to convert to a numeric matrix.

.. as well as an example:

Examples:

```
is.matrix(as.matrix(1:10))
data(warpbreaks)
!is.matrix(warpbreaks) # data.frame, NOT matrix!
warpbreaks[1:10,]
as.matrix(warpbreaks[1:10,]) #using
  as.matrix.data.frame(.) method
```

8

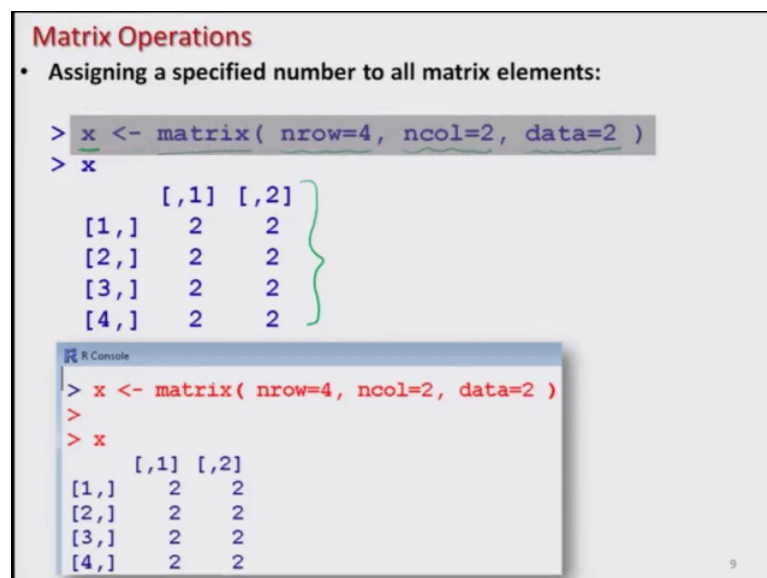
And, similarly they also try to give you here some information that from, where they have taken such information from which source which book and so on. And after that they are also trying to give you some examples. And all these things I have simply copied from the website, wherever this help matrix object has taken us.

So, you can see us that whenever you need a help on matrix. The best approach is to simply type help and inside the bracket within inverted commas type matrix and you will get the most authentic detailed information about the object matrix well. I am trying to

give you here an elementary introduction if you want to read more, I would request you that you try to read all these things once with the once yourself, that will definitely give you more help.

Now I come on some other aspect of matrix operation. Well, up to now what I have done? That I have taken a set of data that was containing the value 1 to 1, and then I have try to arrange it in a form of a 4 by 2 matrix, 2 by 4 matrix or say row wise or say column wise, but now I try to take some more example to it to give you some more idea, that what type of entries I can meet. Suppose I want to have a matrix of order 4 by 2, and I simply want that all the values should be the same, for example 2. So, in simple words I need a matrix of order 4 by 2 in which all the entries are only 2.

(Refer Slide Time: 11:30)



```
Matrix Operations
• Assigning a specified number to all matrix elements:

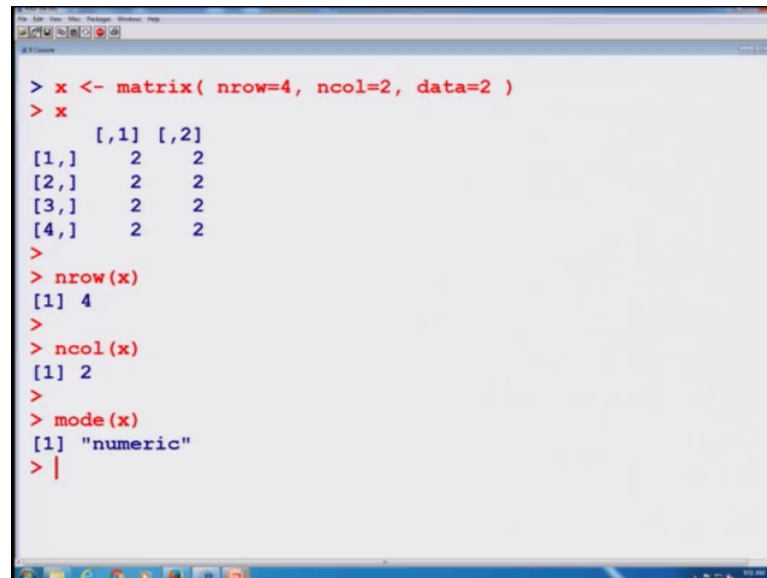
> x <- matrix( nrow=4, ncol=2, data=2 )
> x
      [,1] [,2]
[1,]    2    2
[2,]    2    2
[3,]    2    2
[4,]    2    2

R Console
> x <- matrix( nrow=4, ncol=2, data=2 )
>
> x
      [,1] [,2]
[1,]    2    2
[2,]    2    2
[3,]    2    2
[4,]    2    2
```

So, this thing I can show you here for example, here you can see here x, I am trying to define a matrix, and then matrix and then I am trying to find here the number of rows and then I am trying to define here the number of columns, which are here number of rows here are 4, number of columns here are 2. And, now I read the data which is only 2 in all the entries.

So, I will try to write down here data is equal to 2. And as soon as I do so, I get here this type of output. So, let us try to see it here ourselves. I try to copy and paste this command over the R console.

(Refer Slide Time: 12:07)



```
> x <- matrix( nrow=4, ncol=2, data=2 )
> x
      [,1] [,2]
[1,]    2    2
[2,]    2    2
[3,]    2    2
[4,]    2    2
>
> nrow(x)
[1] 4
>
> ncol(x)
[1] 2
>
> mode(x)
[1] "numeric"
> |
```

And you can see here as soon as you try to do here, you get the outcome here where all the values are here 2, 2, 2, and 2 right. So, this gives you a more confidence that well things are really working. Similarly if you want to know, what is the number of rows of matrix x you can see here there are 4 rows. Similarly if you want to have here number of columns, I would try to find out here number of columns by n col x you can see here this is 2 and similarly if I try to find out here, what is mode of x. Well, that we know that 2 is a numerical value and that is contained in this matrix.

So, you can see here whatever the screenshot I have given, whatever the values I am trying to connote it here they are simply obtained after doing these calculations over the R. Well, I have taken it because it will take a very long time for me, if I start showing you all the operations. And as I said earlier my objective is to help you in learning the R not to teach you R.

(Refer Slide Time: 13:20)

Matrix Operations

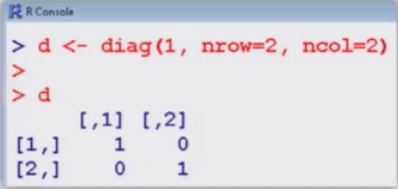
- Construction of a diagonal matrix, here the identity matrix of a dimension 2:

```
> d <- diag(1, nrow=2, ncol=2) Identity matrix
```

```
> d
```

	[,1]	[,2]
[1,]	1	0
[2,]	0	1

$d = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$



10

Now, let us try to consider another aspect. Suppose, I want to construct a diagonal matrix, what is a diagonal matrix? The diagonal matrix is the one, in which all the elements on the diagonals are going to be the same. And off diagonal elements are going to be 0. For example, an identity matrix, identity matrix is a very popular diagonal matrix in which all the elements on the diagonal are 1 and all the elements on off diagonals are 0.

So, if I want to make a create here a diagonal matrix so I can use here that command which is a built in command `diag`; that means, diagonal and inside the bracket, I have to write what are the values which I want on the diagonal. So, here I want to create an identity matrix. So, this is a command for the identity matrix. So, you can see here. So, all the values on the diagonal are going to be one and I need a and identity matrix of order 2; that means, there are going to be 2 rows and 2 columns. The identity matrix is always a square matrix right. So, if I try to do so here. So, I define here number of rows to be 2 number of columns to be two and as soon as I define it I get here a value here like this. And this is trying to show us that this matrix comes out to be like this,

and this is working, but an identity matrix of order 2 by 2. And here I have taken a screenshot here which is trying to give you the things what are happening in R. Now suppose I have a different question that I want to create a diagonal matrix where the values are here something else beside one.

(Refer Slide Time: 15:13)

```
> d <- diag(4, nrow=2, ncol=2)
> d
      [,1] [,2]
[1,]    4    0
[2,]    0    4
> |
```

So, I try to suppose, I want to have here the values suppose 4, so I can make it here like this and then you can see here, that the values on the diagonal they are here 4 and the values on the off diagonal they are here 0. So, this is the way by which I can create a diagonal matrix.

(Refer Slide Time: 15:36)

• Transpose of a matrix X: X'

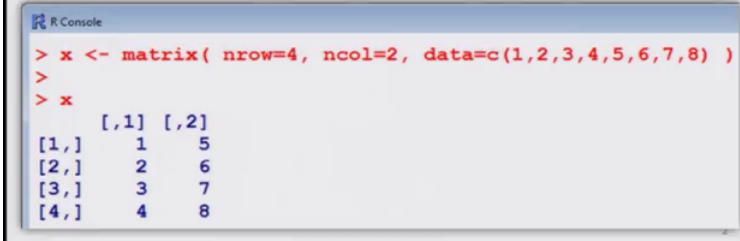
```
> x <- matrix(nrow=4, ncol=2, data=1:8, byrow=T)
> x
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
```

```
> x <- matrix(nrow=4, ncol=2, data=1:8, byrow=T)
>
> x
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
```

Next, thing which I would like to tell you is the following: you see earlier if you remember, I had created a matrix x which was containing the data values 1 to 8.

(Refer Slide Time: 15:59)

```
In R, a 4 × 2-matrix X can be created with a following command:  
> x <- matrix( nrow=4, ncol=2, data=c(1,2,3,4,5,6,7,8) )  
> x  
      [,1] [,2]  
[1,]    1    5  
[2,]    2    6  
[3,]    3    7  
[4,]    4    8
```



```
R Console  
> x <- matrix( nrow=4, ncol=2, data=c(1,2,3,4,5,6,7,8) )  
>  
> x  
      [,1] [,2]  
[1,]    1    5  
[2,]    2    6  
[3,]    3    7  
[4,]    4    8
```

I can show you where you did it. You see; here I had taken this matrix where the number of rows are here 4, number of columns here are 2 and the data values are here are the integers from 1 to 8; 1, 2, 3, 4, 5, 6, 7, 8. Now, I am trying to show you another way by which I can rewrite this data section briefly; in say using other command right.

So, if you try to see here, again I am trying to create a 4 by 2 matrix here, which is here in which the number of rows are here 4, number of columns are here 2 and the data values are given by here 1 colon 8. I am not writing here 1, 2, 3, 4, 5, 6, 7 8 what I am writing here 1 colon 8, this is another way of giving the input, when I want all the numbers in a sequence. Well, I will be discussing these things later on, but here I wanted to show you that the same matrix I can obtain by a different command and then I am saying here by row is equal to T; that means, the data has to be entered by row and this is true T means true. So, once I try to do so, I get here the matrix the same matrix which you had obtained; means this is 1, 2, 3, 4, 5, 6, 7 and 8, because the data has been arranged by row ok.

Now, on the same operation I will try to give you another idea, how to obtain the transpose of a matrix? What is the transpose of a matrix? A matrix has certain number of rows and columns. When I try to interchange the rows and column; that means, rows becomes columns and column become rows, then it is called as a transpose of a matrix. So, how to obtain the transpose of a matrix, this is what I want to show you here.

(Refer Slide Time: 18:10)

• **Transpose of a matrix X: X^t** X' X^T

```
> xt <- t(x)
```

```
> xt
```

	[,1]	[,2]	[,3]	[,4]
[1,]	1	3	5	7
[2,]	2	4	6	8

$x = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \end{pmatrix}$
4 x 2

2 x 4

```
R Console
```

```
> xt <- t(x)
```

```
> xt
```

	[,1]	[,2]	[,3]	[,4]
[1,]	1	3	5	7
[2,]	2	4	6	8

12

So, I am trying to consider here this matrix and this is a screenshot of the matrix. And now the usual symbol in mathematics to denote the transpose is x something like prime which is called as x transpose or in some books they try to write down here x and on the superscript capital t which means x transpose. In order to obtain the transpose of a matrix in R the command here is `t` and inside the bracket you have to write down the matrix or the variable that is containing the matrix. So for example, here I have defined a matrix x . So, I would like to find out that the transpose of a matrix x . So, I try to write down here `t(x)`, `t` inside the bracket I try to write down the variable name x . And this transpose of x I am trying to store in another variable say xt . This I am trying to do so that we can use it later on also. So, as soon as you try to do it here you can see here that xt comes out to be like this.

So, earlier your x was 1, 2, 3, 4, 5, 6, 7, 8 and now what is happening? There are 4 rows and there are 2 columns, but now there are here there are 2 rows and 4 columns and whatever are my values here 1, 3, 5, 7 they become here 1, 3, 5, 7. First column becomes first row and second column 5, 2, 4, 6, 8 becomes here second row 2, 4, 6 and here 8. Let us try to do this operation of ourselves over the R. so that I can show you here. So, first I try to create here this matrix here. So, you can see here this is my here matrix x and now I try to find out the transpose of this matrix here x .

(Refer Slide Time: 20:21)

```
> x <- matrix(nrow=4, ncol=2, data=1:8, byrow=T)
> x
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
>
> t(x)
      [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
> |
```

So, this comes out to be like this. So, this is how you can find out the transpose of a matrix. So, I would request you that you try to take some more examples and try to do all such operation useful on your computer. So, that you become more confident about using these things ok.

Now, I come on say another aspect, how to do the multiplication? There are certain rules of multiplications in matrix theory.

(Refer Slide Time: 21:07)

• **Multiplication of a matrix with a constant**

```
> x <- matrix(nrow=4, ncol=2, data=1:8, byrow=T)
> x
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
```

A B.
1 2 3 4
5 6 7 8
x 2

```
R Console
> x <- matrix(nrow=4, ncol=2, data=1:8, byrow=T)
>
> x
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
> |
```

13

For example, whenever I am trying to multiply here two matrices say here a and b. Then suppose a has an order of say here p cross q, then we should have a order q cross say either q cross q or q cross r such that these two values should be the same. That is what we have to keep in mind. Many times you will see that whenever you are trying to write a program, sometimes it gives you a very popular error message that the dimensions are not matching in R. So dimensions are not matching; that means there is some matrix whose product has been defined in such a way. So, that the number of columns of the first matrix are not matching with the number of rows of the second matrix, is a very common mistakes what would be encountered while doing the programming. So, again I will try to consider here the same matrix which I had considered earlier right.

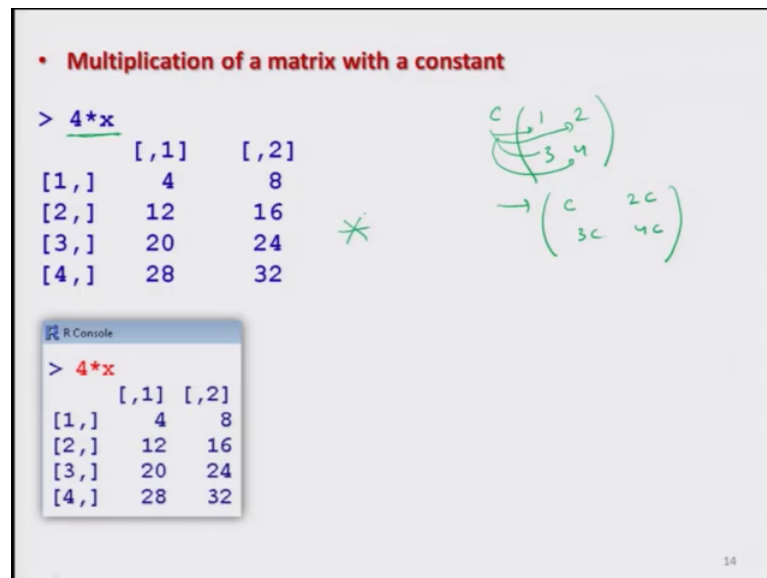
(Refer Slide Time: 22:04)

• **Multiplication of a matrix with a constant**

```
> 4*x
      [,1] [,2]
[1,]    4    8
[2,]   12   16
[3,]   20   24
[4,]   28   32
```

\times

$c \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \rightarrow \begin{pmatrix} c & 2c \\ 3c & 4c \end{pmatrix}$



R Console

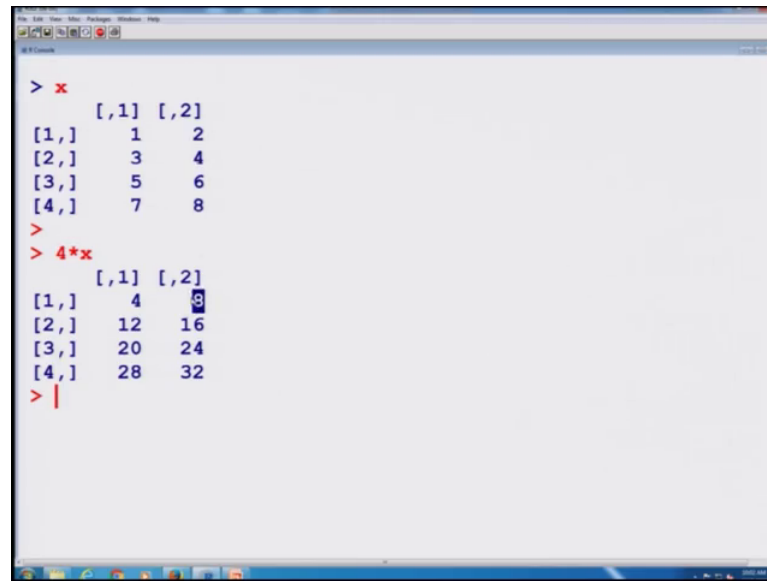
```
> 4*x
      [,1] [,2]
[1,]    4    8
[2,]   12   16
[3,]   20   24
[4,]   28   32
```

14

And now in order to do a matrix multiplication, what we see that? Suppose I am trying to multiply a matrix by a constant. Some say some constant and then multiplied by here some matrix here 1, 2, 3, 4 then what happens that we know from the rules of mathematics, then whenever we are trying to multiply it this c is multiplied with each of this entries.

So, this becomes here c, 2 c, 3 c and here 4 c. So, suppose if I try to multiply here see here this matrix x by 4 then; that means, each element of the x matrix will be multiplied by here 4. Let us try to see it here. So, for example, here you have seen there is here an x I can clear it.

(Refer Slide Time: 22:58)



```
> x
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
>
> 4*x
     [,1] [,2]
[1,]    4    8
[2,]   12   16
[3,]   20   24
[4,]   28   32
> |
```

So, that you can see more clearly x and now if I try to star say 4 into x. So, you can see here because element is multiplied by 4 from here this multipliers and this element here is two in the matrix x and now when you try to multiply it this becomes here eight. So, you can see here that all the elements are being multiplied by the constant here for. So, this is how you try to do the matrix multiplication by a constant. One thing what you have to keep in mind here, that whenever you are trying to multiply a matrix by a constant, then your operator here is say star simple multiplication sign.

But, later on you will see that whenever we are trying to multiply two matrices, then the operator will be different that does not remain as say star right. So, I would say that I have given you a start up with some matrix operation and I would request you that you please try to do some practice with these simple operations. And in the next lecture I will continue with some more operations on the matrix theory. And it is important for us to practice it more, because these things looks very very simple, but you have to keep in mind one thing that whenever something is happening inside a computer you cannot see it. You simply have to understand the basic fundamentals by which you can only know that whenever you are trying to give a command on the R console, then how the output is coming out. R will not come to tell you that how I am working, but this is only you who has to understand that what operations have been made on the values. So, that I am getting these values. So, we will see in the next lecture till then goodbye.