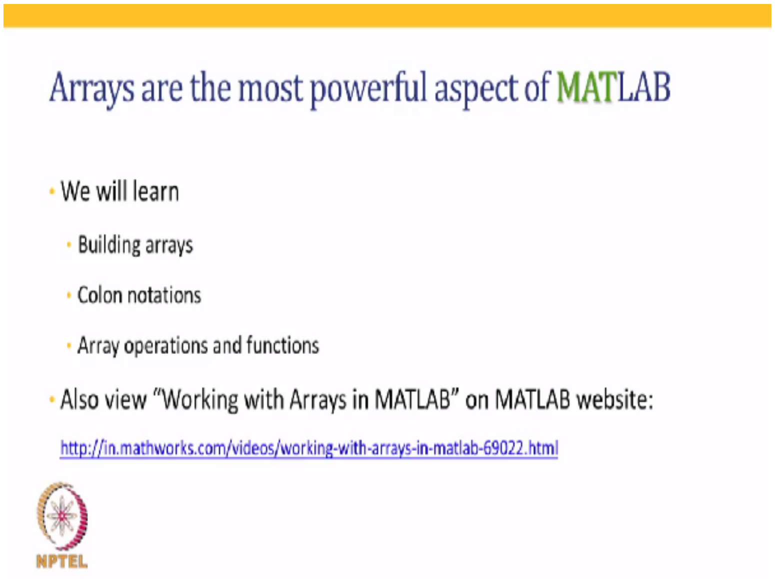
MATLAB Programming for Numerical Computation
Dr. Niket Kaisare
Department of Chemical Engineering
Indian Institute of Technology, Madras

Module No. # 01
Lecture No. #1.2

Introduction to MATLAB programming – Arrays and Matrix Operations In MATLAB

Hello and welcome to the MATLAB programming NPTEL course, we are in module 1 which is introduction to MATLAB. Lecture 1.2 is going to cover arrays and matrix operations in MATLAB. In the previous lecture we had given an overview of what MATLAB is.

(Refer Slide Time 00:29)



Arrays are the most powerful aspect of MATLAB

- We will learn
  - Building arrays
  - Colon notations
  - Array operations and functions
- Also view "Working with Arrays in MATLAB" on MATLAB website:
  http://in.mathworks.com/videos/working-with-arrays-in-matlab-69022.html

In this particular lecture, we will learn how to build arrays which we had started off in the previous lecture itself. We will go over that again. We will talk about colon notations and we will talk about array operations and array functions. As before there is video in, on the MATLAB website that you can watch the link for which is provided over here.

The video is called working with arrays in MATLAB. And I suggest that you also look at this video in addition to this NPTEL lecture okay. So let us look at building of MATLAB arrays and vectors okay.

(Video Starts: 01:09) In the previous lecture, we had look at how to build row vector. If we have row separated, elements of the row separated by commas, we are going to have a row vector and if instead we were to separate by semicolon, we are going to have a column vector as shown over here okay. So, I am getting an error the reason why I get an error is, because I forgot to put a semicolon over here.

So, it is trying to build a vector by having a 1 in the first row and in the second row, we have 2 elements and we cannot build a vector of that size, so instead if we were to have semicolons as shown over here we are going to get a column vector as seen over here. That leads me to next thing and that is, that we do not have to build vectors and arrays by having each and individual elements we can use previously defined vectors and arrays to build a particular matrix also.
So, let us say we were to define c as part, as let say identity matrix of size 3 okay. So, we will get c as eye 3, eye is command in MATLAB that will build an identity matrix okay. So, what we have over here is a vector a that is a 3/1 sorry, 1/ 3 vector and a vector c or a matrix c that is a 3/3. So, if we write say d=a;c. This is the result that we are going to get. So the first row is what our vector a was. And next 3 rows are going to be what our c the array c was.

So, this is how we have built our matrix d. Let us say we try to do this a, c what is going to happen is, we are trying to build a matrix which by combining an array which has 1 row with a matrix that has 3 rows which is going to result in an error, the same similar type of error as we had seen earlier. This is an error in horizontal catenation that means we are combining 2 arrays horizontally and there is an error in combining that.

Recall earlier that we had what is known as a vertical catenation error. However instead if we were to give e = b, c, that is going to work because this is our vector b which has 3 rows and we are horizontally catenating it with another vector, that has another matrix sorry, that has 3 rows and we will be able to do that.

The next function is transpose of a vector or a matrix. And for transpose you have to use apostrophe, so or dash. So, if you we say a dash we are going to get instead of a column vector instead of a row vector we are going to get a column vector. And just as we did before we can construct a matrix now saying a’, c.

Now a dash what is going to happen is first we are going to have transposed. So, first MATLAB is going to calculate the transpose of a. a recall was a 1/3 array. So if we transpose the 1/3 array we are going to get a 3 /1 array, so now 3/1 array can be horizontally catenated with a 3/3 array why because, (Video End 5:16)

We have the same number of rows in a prime as well as in c. So, if we are giving this particular command as you will see later.

(Video Starts 05:24) We will see that we are able to construct the matrix f. So, f = a`,c enclose in square brackets, press enter and we are going to get our matrix f okay. Now let us say we wanted to get any particular element of this matrix.
So let us say we wanted to get this particular guide 2 over here. This is row number 2 and column number 1 so we can get this by saying f not square but round brackets f (2, 1). So, that is row number 2, column 1 and if we press enter, we are going to get this particular element.

Likewise, let us say we wanted to get this element, that is now in f, it is in row number 3 and column number 2 and press enter and we are sorry, row number 2 column number 3. Row number 2 and column number 3 and we are going to get this particular element okay. So when we give this command it is row number 2 the first guy is the row and the third column over here that is column number 3. When we give this command that is the row number that is going to be row number 3 and column number 2 okay. So that is what we are going to get.

Now if we were to give let us say f (5, 5), now as you know that f (5,5) does not exist because we do not have 5th row nor we have 5th column, if we give this, we get an error it says index exceeds matrix dimension. So that is because that the size of the matrix was 3 /4 and we were trying to query f (5, 5) that does not exist okay.

So, I will clear the screen over here. Now if I wanted to find the size of that matrix, there is a very simple command called size. If I give size f, it is going to return to me the size okay. The number of rows is 3 and number of columns is 4. (Video Ends 07:40)

(Refer Slide Time 07:41) Okay so, that is about building the arrays. I will next talk about colon notations. So, colon notations is so colon in MATLAB is nothing but 2 or say want to go from say 1 to 5 we use a colon 1:5 and let us look that here.

(Video Starts 7:59) If I give 1:5, I have 1, 2, 3, 4 and 5. Note that when we use colon notations MATLAB creates a row vector and not a column vector. We can always assign this to any other variable, so let us say we were to say z = 1 to 5 this is what we are going to get. Now let us say we wanted to have 1 to 5 in steps of 3 we will give this as y = 1:3:5 okay.

So, in steps of 3 means the first guy is 1, second guy is 1+3 that is 4 and the third guy would be 1+ 4 + 3 that is 7 ,however 7 goes beyond the value 5 so 7 will not be admissible over here. So, if I press enter, I will get 1 and 4 as the 2 elements in this particular vector. Now if I were to do y = let us say 2: 1, what is going to happen ok.

Now 2:1 means we want to go from 2 to 1 in steps of 1 the 1st guy itself 2 is greater than 1 and therefore we are not going to get anything we are going to get a blank matrix. Okay so, y becomes an empty matrix which has 1 row and 0 columns. Note that MATLAB is not giving an error; MATLAB is giving an empty matrix okay.

Likewise if you were to have y = [], we are going to get an empty matrix. So that is the 0 / 0 matrix it is an empty matrix okay. So, empty matrices are also matrices in MATLAB of size 0. (Video Ends 09:59)

Keep that in mind because that is an important information okay. (Refer Slide Time 10:04). So, we have kind of covered what colon notations are just as we had seen earlier we can query any part of a vector or an array using elements of that array, likewise we can query.

(Video Starts 10:20). Multiple parts of that array. So let us look at the array f over here. And let us say we wanted to get second row and column number 2 and 3. How we can do that is, f row number 2 and column number 2 and 3 that is 2:3. And we will be able to get 0 and 1 over here. Likewise if we wanted row number 2 and 3 in column 2, we will write f row number 2 to 3 and column number 2.

So, 2:3 means row numbers 2 to 3 and column number 2. Press enter this is what we get that is 0 and 0. Okay, let me clear this screen again okay. In order to query any of the elements or any of the sub matrix, the command that we are going to use is f (p, q), where p represents all the elements that we want to query and in a row and q all the elements that we want query in a column.

So, if you were to say (3, 3) that going to f (3, 3) which is 0. Likewise say p = 0, sorry p = 3, q = 3 and if you were to query f (p, q), we are also going to get f (3, 3) okay. Now instead if you were to say p is 1,2 and if you say f is p, q what is that going to do is, we are going to query the first and second rows of f and the first column of f because p is 1,2 and q is 3 okay.

And this is what we are going to get. So f what we queried is because p was 1, 2. So first and second row q was equal to 3, so column number 3. So, first and second row of column number 3 is first and second row of column number 3 is what we get okay. Likewise we can say f 1 to 2 and 3 to 4 and this is what we will get. So, again let us type f over here so, what we have is f rows number 1 and 2 and columns number 3 and 4. So, we are going to get this particular sub matrix as our ans okay. (Video Ends: 13:01)

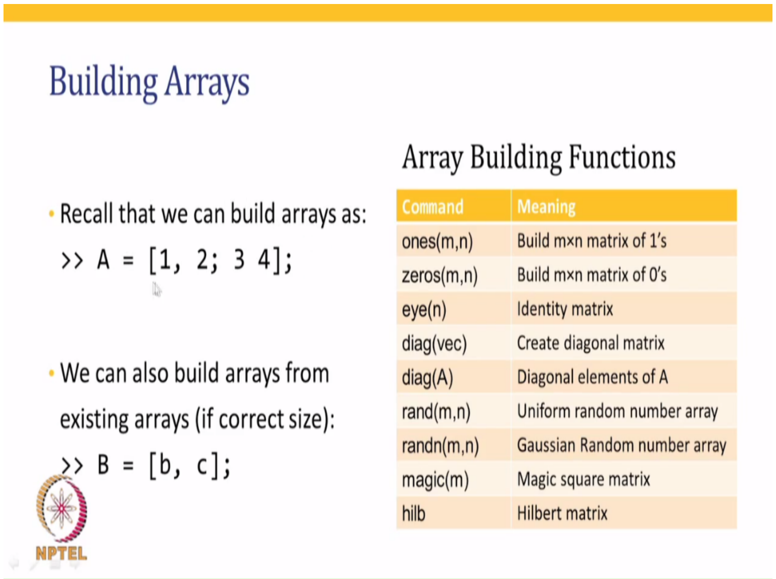
So, what now we have covered so far is, how to build an array and how to use colon notations both to build an array as well as to get parts of an array or sub vectors in an array or a sub matrix or the sub array in that particular array okay.

These are going to be extremely powerful tools that we are going to use and leverage in the rest of this particular course. And I encourage you to look at the videos that are there on the

MATLAB website as well. As well as try these things out yourself at the end of this particular module. We will have a certain set of examples that you can try.

These examples will be split based on the various lectures and I would encourage you to start going through this particular examples right away in parallel to doing this video lecture okay. So let us go to the next slide Okay.

(Refer Slide Time: 14:00)



Building Arrays

- Recall that we can build arrays as:

```
>> A = [1, 2; 3 4];
```

- We can also build arrays from existing arrays (if correct size):

```
>> B = [b, c];
```

Array Building Functions

Command	Meaning
ones(m,n)	Build m×n matrix of 1's
zeros(m,n)	Build m×n matrix of 0's
eye(n)	Identity matrix
diag(vec)	Create diagonal matrix
diag(A)	Diagonal elements of A
rand(m,n)	Uniform random number array
randn(m,n)	Gaussian Random number array
magic(m)	Magic square matrix
hilb	Hilbert matrix

This is what we have use in order to build the arrays. We use the comma and semicolon in order to build an array using numbers as we are shown over here. In addition to that, we can also use existing arrays if there are of an, of a correct size in order to build arrays as well okay. In addition we have also seen in this function eye that is to create n/n identity matrix. In addition to that there are several other array building functions such as ones, zeros, diag, rand, randn, and magic, Hilbert and so on and so forth.

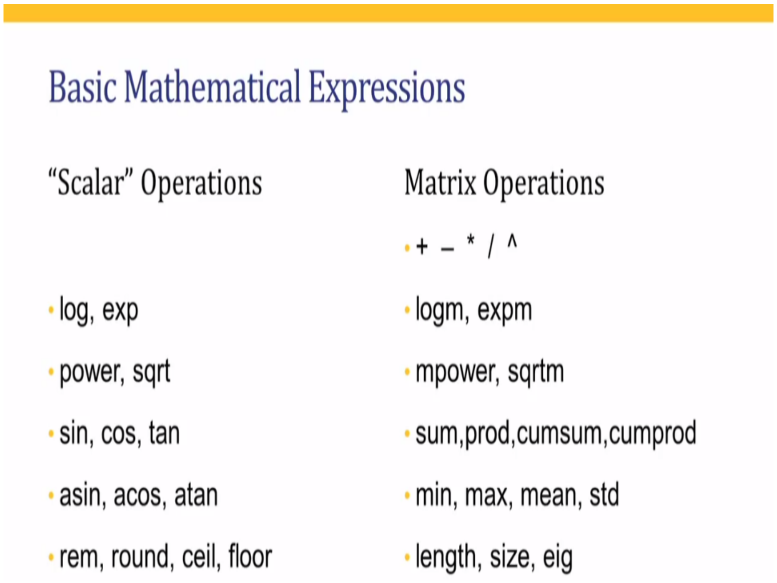
(Video Starts 14:42) So, I will quickly show you, how to build, how use ones. If we give the command ones 2, 4, we are going to get an array containing all ones of 2 rows and 4 columns. This is what we see over here and that will be assign to a variable ans okay. If I give the command say x, capital x equal to zeros 2, 1, we are going to get 2 rows and a single column with the el all of the elements getting the value 0, so that is what we matrix functions are going to do. (Video Ends 15:23)

So let us look at another function called rand. Rand is basically uniformly distributed random numbers.

(Video Starts: 15:29) So, if I give rand without any arguments, I am going to a single random number. If I give rand with 2 arguments 2, 3, I am going to get a 2/3 matrix of random numbers, these are the uniformly distributed random numbers that means you will get values between 0 and 1 with equal probability okay. (Video Ends: 15:59).

So these are the various array building functions.

(Refer Slide Time 16:06)



Basic Mathematical Expressions

"Scalar" Operations	Matrix Operations
	+ – * / ^
log, exp	logm, expm
power, sqrt	mpower, sqrtm
sin, cos, tan	sum,prod,cumsum,cumprod
asin, acos, atan	min, max, mean, std
rem, round, ceil, floor	length, size, eig

We also saw in the previous lecture some of the basic mathematical expressions. Scalar expressions where a logarithm, exponent, power, sin, cos and so on. There are matrix operations +,-,*, /. These are not scalar operations but matrix operations. So what I am mean by that is,

(Video Starts: 16:29) that we need our 2 vectors to be commutated. So let us bring out our vector a okay. If I do a * a, we are going to get an error that is because a 1/3 matrix cannot be multiplied by a 1/3 matrix. The number of columns of a has to be equal to the number of rows of the one that you are multiplying that a with. So if you give a, we get an error saying that inner matrix dimension must agree. The reason why that happens is, a is a 1/3 vector.

Let us say we have a vector b. Now b was a 3/1 vector. So, we can indeed multiply a 1/ 3 vector with a 3/ 1 vector. So, you can indeed do a*b and that is going to be a matrix multiplication okay. Let us look at the vector the matrix e. e was a 3 / 4 matrix where as a is 1/3 matrix. So, indeed we can multiply a with e. So, if we do a multiply by e, we are going to get 1/ 4 vector okay. And this is what we get. When we do a multiplied by e, again the rules of matrix multiplications are indeed available or applicable over here okay. (Video Ends: 17:52).

The slash, we will not go over this slash currently that is because a little bit more advanced at this stage. This slash is going to be either depending on what slash you use either it is going to be left division or a right division. What left division and right division means something that we will go in one of the later modules.

We are not going to cover that in this particular module. If you are going to use a slash use that with care and wait until the next module that is going to cover the slash operation. The next one is a power operation and that operation is the caret sign.

(Video Starts: 18:39). The caret sign is again a matrix operand. So we cannot use that caret for a non square matrix that has to be use on a square matrix itself. So, if we do a caret 2, it is same as a * a. These are the same commands and as we had seen a is not commutative towards self under multiplication. However instead if we had a 3/ 3 matrix let us say r = rand (3, 3) okay. That would be commutative in under power as well. So, r to the power 2 is just going to be r*r. So it is a matrix multiplication and this is what we will get okay. (Video Ends: 19:28)

What I am showing over here also are the scalar operations and equivalent vector operations. So here if we do exp of an array, we are going to get element by element exponential. If we want a matrix exponential, we want to do expm.

(Video Starts: 19:50). Okay let us look at that. Let us say if we had say a matrix d = 1, 0, 2, 1 and if we were to do, let me just clear this clc, okay 1 0 2 1. If was to do exp of d I am going to get each element going to get exp its exponent. If instead if I were to do exp m of d, this is what I going to get and this is a matrix exponent. And matrix exponent has its own rules something that we are not going to go over in this particular module. But this is an individual element by element exponent and this is a matrix exponent. And as you can see that the matrix exponent is different than element by element exponent. (Video Ends: 20:48).

Okay in addition to that there are other commands sum, prod, cumsum and cumprod. Let us look at those commands also. (Video Starts: 20:58). So our matrix a was 1, 2 and 5, sum is going to be just sum of the elements of that particular matrix so 1 +2 +5. Cumulative sum is cumsum. Here if you give cumsum of a, what we are going to get is, first element is going to itself the second element is going to be 1+2 the third element is going to be 1 +2 +5 okay. So we get 1, 3 and 8 as the result of cumsum.

Likewise we have prod, prod is just multiplication of all the elements of a and cumprod is multiplication of all the elements until that particular. so cumprod of a is 1 then 1 multiply by 2 and then 1 multiply by 2 multiply by 5 and this is a result that we get. (Video Ends: 21:52)

Okay, this command size we have already seen. Length is a length of a particular array. So, if we have an array which is say 5/1 and if we give command length, it will return the value of 5 okay. Again these are things that you should try out in MATLAB okay.

Now in addition to the matrix operands we also have an element by element operand. Recall the difference between log and logm. If we wanted to do this with multiplication, division and power sign, we have an operand called dot star, dot slash and dot caret which will operate on each element. And let us look at that (Video Starts: 22:37)

Let us define our variable a as 1, 2, 3, 1 okay. If I do a*a, we are going to get a which is going to be multiplied by itself. That is going to be matrix multiplication. However, if I am going to do a .*a, it is going to be each element of a multiplied by itself. So we are going to get 1 multiplied by 1 that is 1, 2 multiplied by 2 that is 4, 9 and 1 this is what we will see when we do the result over here okay.

Exact same thing is what you can expect with a c ^ 2 okay. And that is going to be same as a * a squaring of a okay. And likewise if you want this similar to a.*a, that is going be a. ^. So, a. ^ 2 is going to give us 1, 4, 9, 1 that means each individual element of the array has been squared okay. (Video Ends: 23:44)

So these are the basic mathematical expressions. Again for people starting out with MATLAB, some of these operations are little bit difficult to grasp. So, I would strongly suggest that you go and try out these operations, take a pen and paper along with you and see what the matrices that you are forming and what results you are going to get. With that I come to the end of this particular lecture and I will see you in lecture 1.3 thank you.