

**Numerical Linear Algebra**  
**Dr. P. N. Agrawal**  
**Department of Mathematics**  
**Indian Institute of Technology, Roorkee**

**Lecture - 18**  
**Introduction to MATLAB**

Hello friends, welcome to this lecture. In this lecture will give some introductory introduction to MATLAB just a basic introduction of MATLAB. And this MATLAB is going to be very, very useful in coming lectures of our course. So, in this course we will just discuss how to open how to do certain calculation based on based on our course in MATLAB. So, let us start our lecture introduction to MATLAB.

(Refer Slide Time: 00:53)

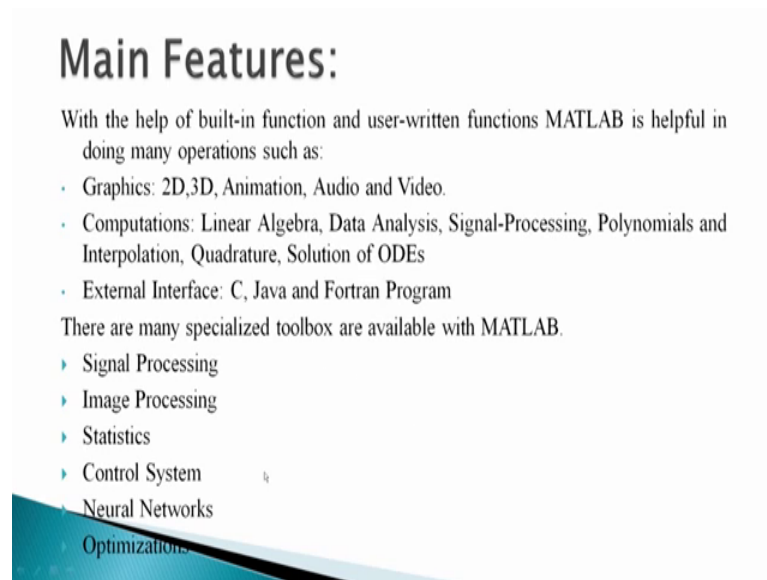
## What is MATLAB

MATLAB (MATrix LABoratory) is a software package for high-performance numerical computation and visualization.

It provides an interactive environment with many of built-in functions for technical computation, graphics, and animation.

So, what is a MATLAB? MATLAB is basically matrix laboratory, and it is a software package for high performance numerical computation and visualization. And it provides an interactive environment with many of the built-in function for technical computation graphics and animation. So, it is a basically a say quite very important tools, but will use only for say our co course as a matrix calculation of based on matrices.

(Refer Slide Time: 01:27)



## Main Features:

With the help of built-in function and user-written functions MATLAB is helpful in doing many operations such as:

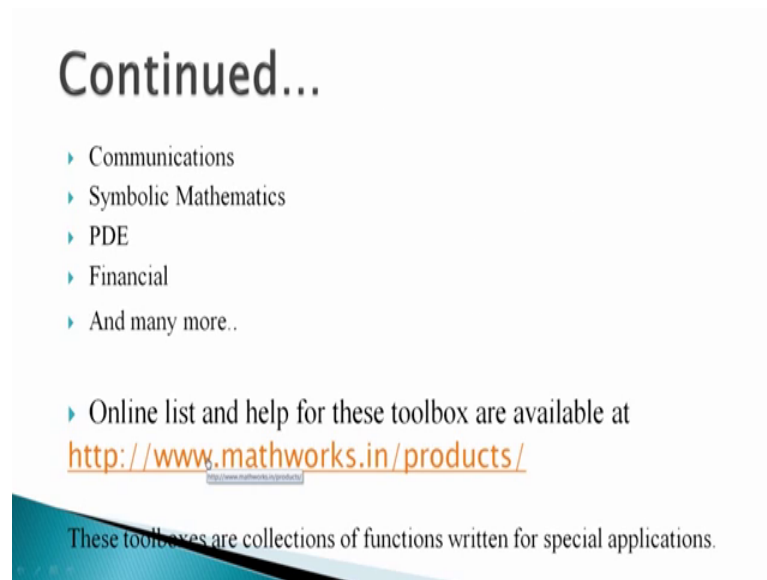
- Graphics: 2D,3D, Animation, Audio and Video.
- Computations: Linear Algebra, Data Analysis, Signal-Processing, Polynomials and Interpolation, Quadrature, Solution of ODEs
- External Interface: C, Java and Fortran Program

There are many specialized toolbox are available with MATLAB.

- ▶ Signal Processing
- ▶ Image Processing
- ▶ Statistics
- ▶ Control System
- ▶ Neural Networks
- ▶ Optimizations

So, main features of MATLAB is that with the help of built in function, and user written functions MATLAB is helpful in doing many operation, and some operations are as follows. In graphics we can plot 2D, 3D, animation audio and video thing. And if computation we can do computation related to linear algebra, data analysis, signal processing, polynomials an interpolation quadrature solution of ODE's. An external interface means, we can take the help of our programming language, such as C java and Fortran program. And based on this we have already many specialized toolbox are available with MATLAB. And these toolboxes are signal processing, image processing, statistic, control system, neural networks, optimization, communications symbolic mathematics, PDE financial and many more.

(Refer Slide Time: 02:23)



**Continued...**

- ▶ Communications
- ▶ Symbolic Mathematics
- ▶ PDE
- ▶ Financial
- ▶ And many more..

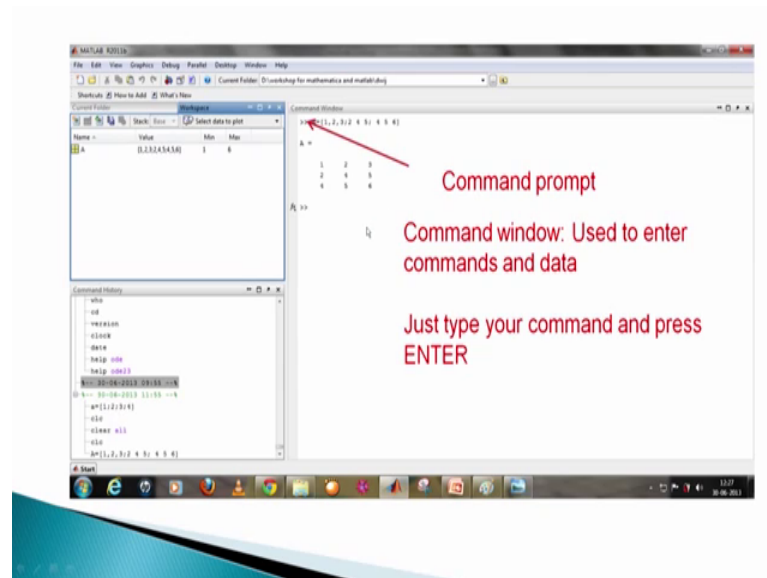
▶ Online list and help for these toolbox are available at  
<http://www.mathworks.in/products/>

These toolboxes are collections of functions written for special applications.

And the complete list for toolbox and help for this tool box are available on this thing; that is, w w dot mathworks dot in slash products.

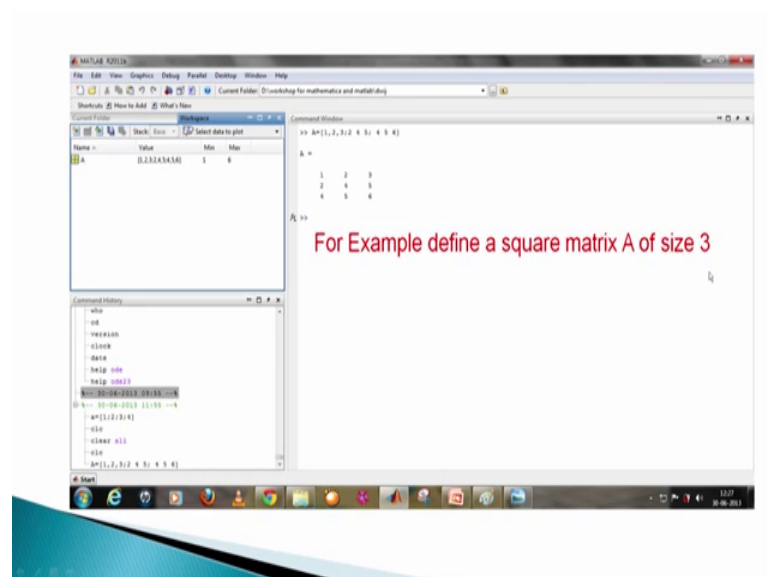
So, here you will get the list of toolbox available. And these toolboxes are nothing but collection of functions written for special application. So, if I look at the toolbox PDE, or say symbolic mathematics, in symbolic mathematics you can do calculation using symbols only. You need not to define numerical values for each symbols. So, that is why these specific toolbox are very, very important. So, once we start our MATLAB and then your window will look like this.

(Refer Slide Time: 03:15)



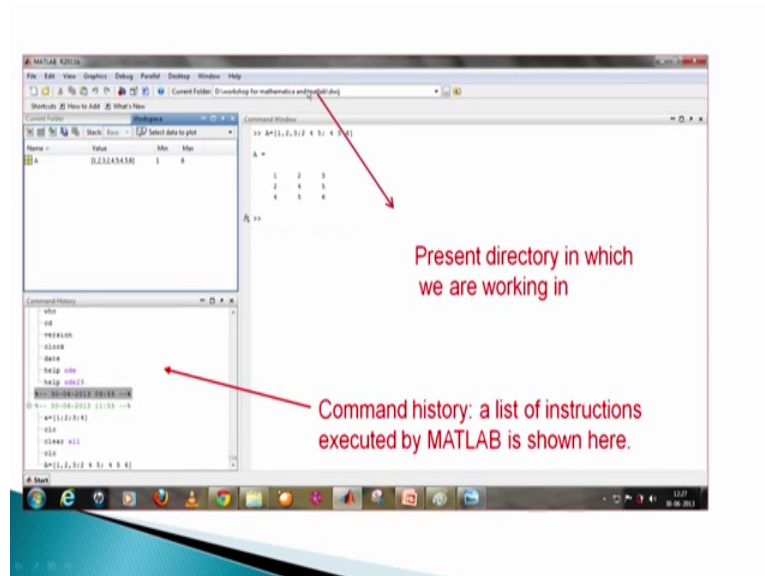
So, here this is a one window which we have just opened. So, here this window, and then this is a command prompt. And command this is known as command window. And this command window is used to enter commands and data. And once we have command written here, just type your command and press enter. Then it will say work with your command and give you the result.

(Refer Slide Time: 03:47)



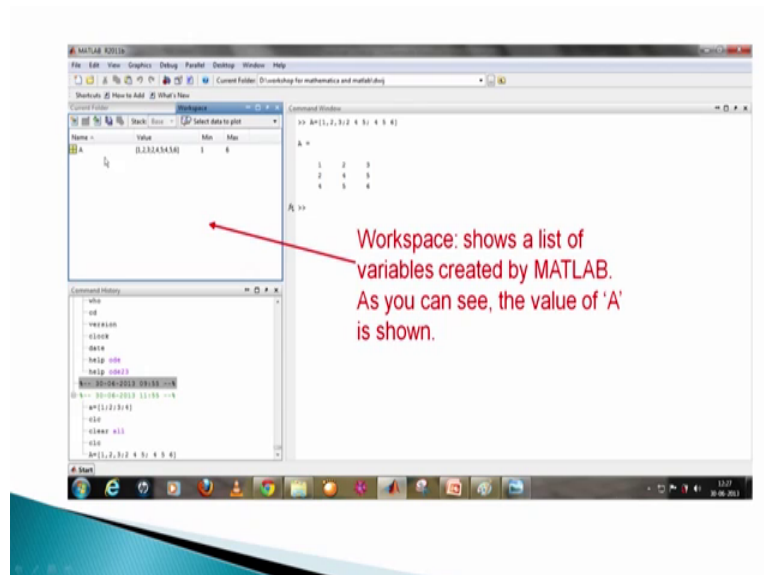
For example, if you want to define a square matrix a of size 3 then we can write A as this 1, 2, 3, call 2, 4, 5, and 4, 5, 6 and when we enter we have this matrix a as written as this.

(Refer Slide Time: 04:04)



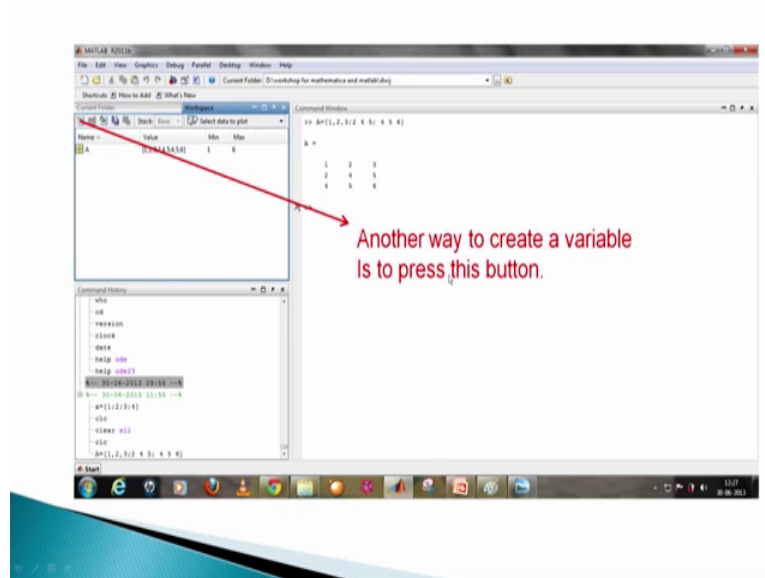
Then this is a the present directory in which we are working in. So, here this current folder will give you the directory in which we are working. And this is the command history, and it will display all the instruction which we have executed by MATLAB is shown here.

(Refer Slide Time: 04:23)



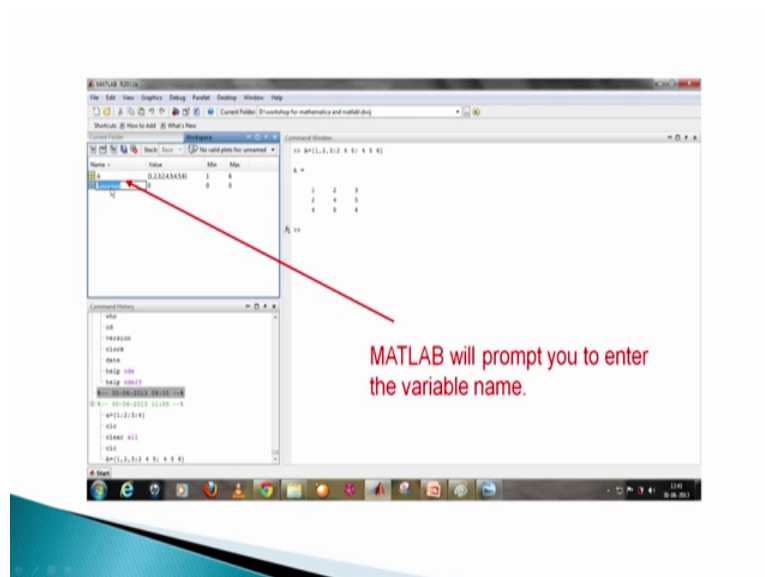
Then here we have workspace. So, workspace, a list of variable created by MATLAB and we can see that this value of a shown here, the name is given which will give you the variable name, and the a will have this value, and here we have minimum max.

(Refer Slide Time: 04:47)



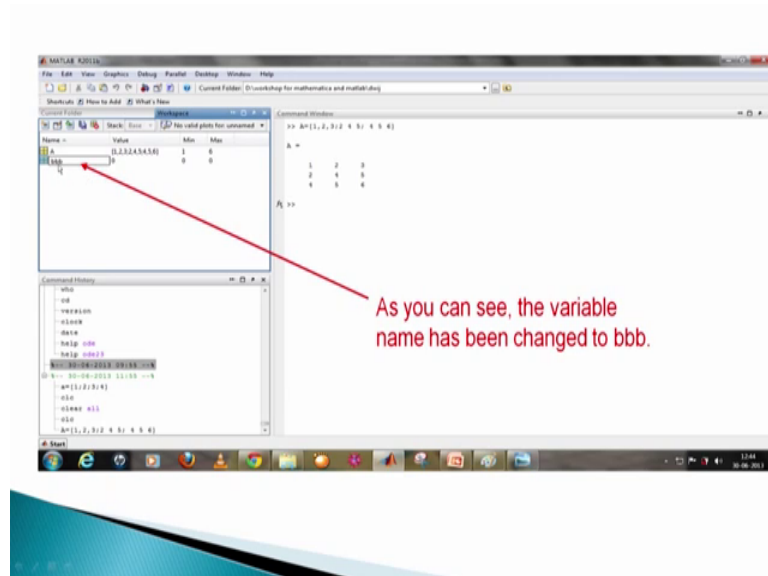
And another way to create a variable is to press either this button. Or we can click here.

(Refer Slide Time: 04:53)



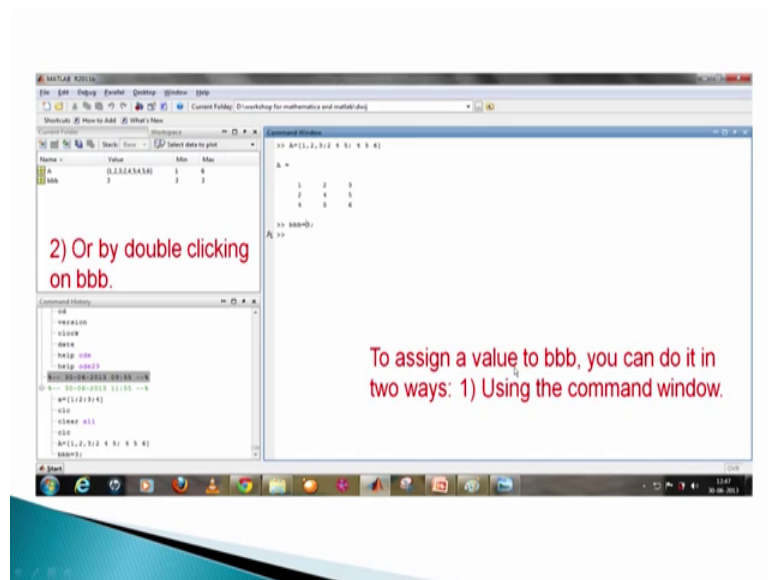
And MATLAB will prompt to you to enter the variable name. So, here you click and you will get this. And here you can give the variable name.

(Refer Slide Time: 05:04)



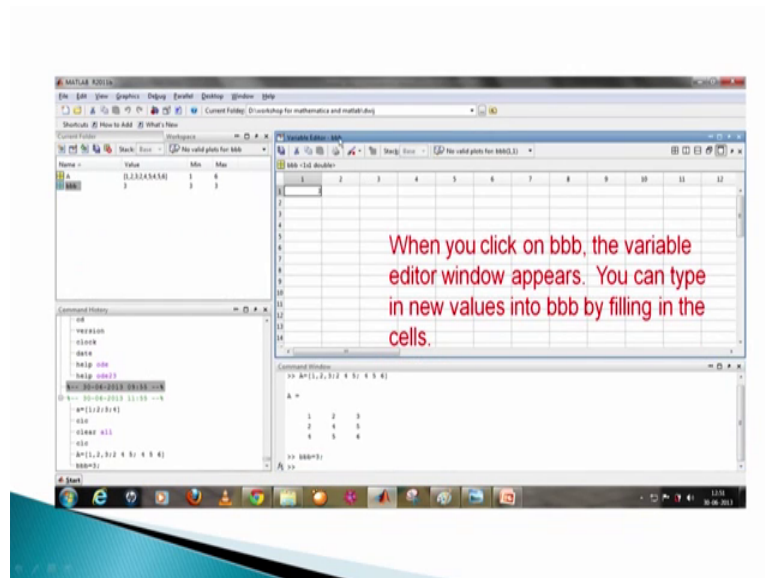
For example, if I want to create b b b. So, we simply click here and we ~~didn't~~ did not b b b, and then we try to show the so, we can see that the variable name has been changed to b b b.

(Refer Slide Time: 05:18)



And to assign a value to b b b, you can do it in 2 ways. One way is to use the command window. So, here you write down the variable name equal to and we want to assign some value suppose we want to assign the value 3. Then it will be stored here in the variable name, or by double clicking on this.

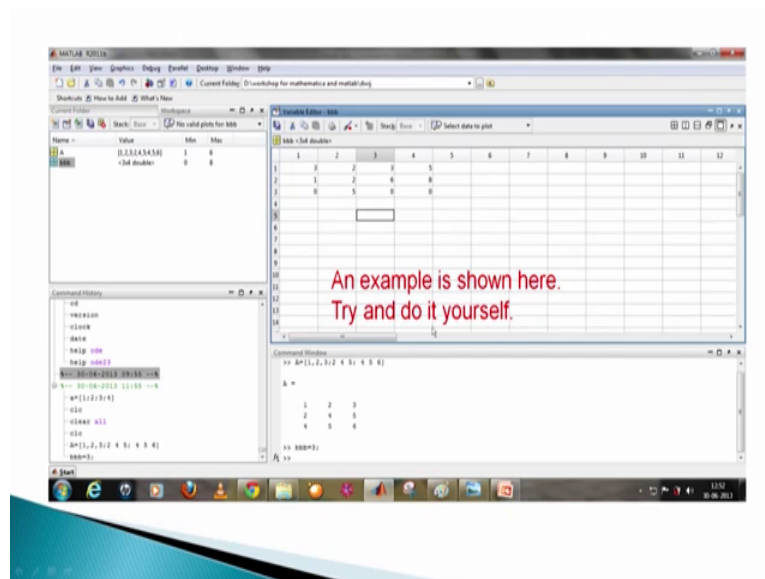
(Refer Slide Time: 05:42)



So, you just double click on this, when you double click it this then we have this editor.

So, when you click on b b b the variable editor window appears. And you can type in new values into b b b by filling in these cells. So, once we have this you can click here and you can say and give anywhere. Any value to this variable triple b and an example is shown here and try and do it yourself.

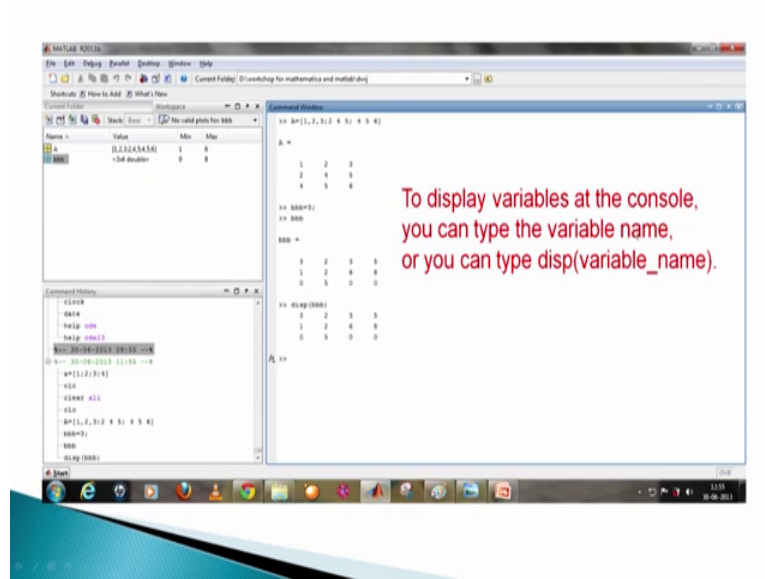
(Refer Slide Time: 06:04)



So, here we have just listed some values here 3 2 3 5 and some values given here.

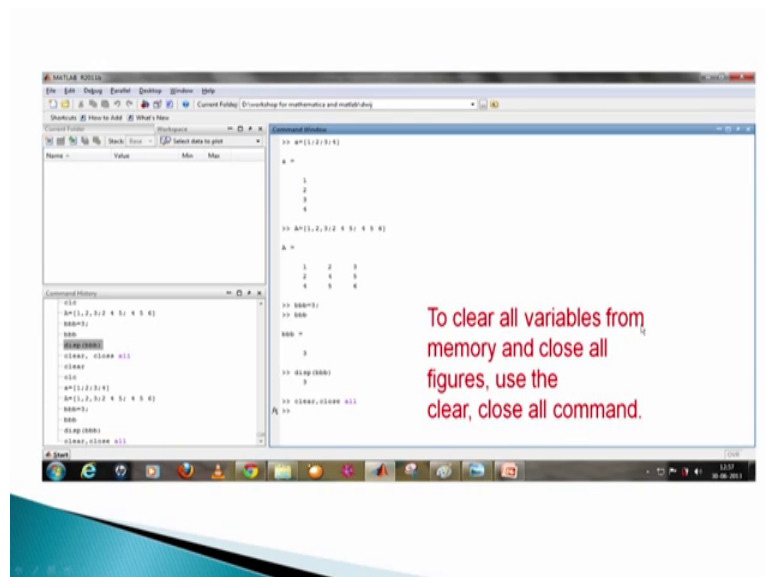


(Refer Slide Time: 06:16)



So, you may try on your own. So, do and once it is created or when values are assigned, then we can display variable at the console and you can type the variable name or you can type display. So, they are 2 way to display this either you simply write the variable name it will give you the variable listed here, or you simply display the variable. So, this variable name and it will give you the. In fact, these 2 are the same result basically.

(Refer Slide Time: 06:47)

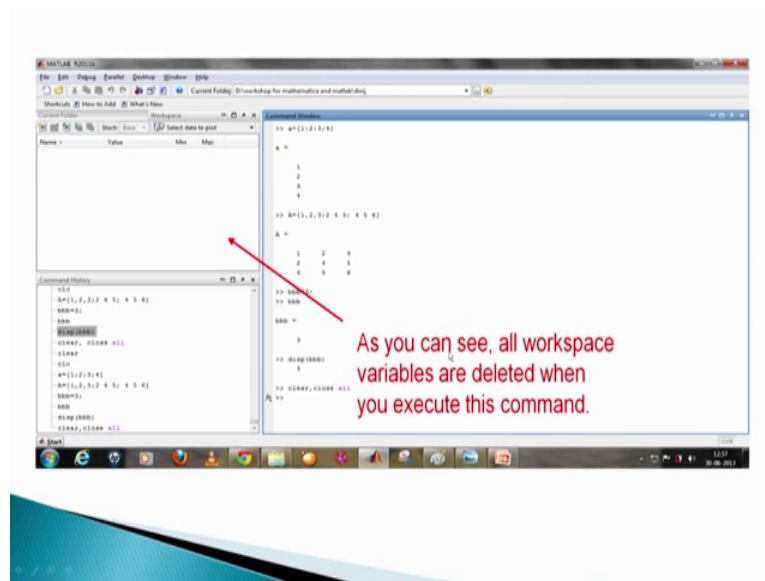


And to clear all variables from memory and close all figures use the `clear` and `close all` comment. So, here we use `clear` and `close all` that will erase all the variables saved here.

The reason using this command is that sometimes when we do say long process and these variable names are say quite messy kind of thing. Suppose, we have started with some matrix a and we are doing some kind of calculation. And then when we may be say a x y all these variable names we have assigned. And when we start a new column then again when you write any matrix then it will take the latest say value assigned to that variable. So, it is always say advisable to clear all variable before starting any new job.

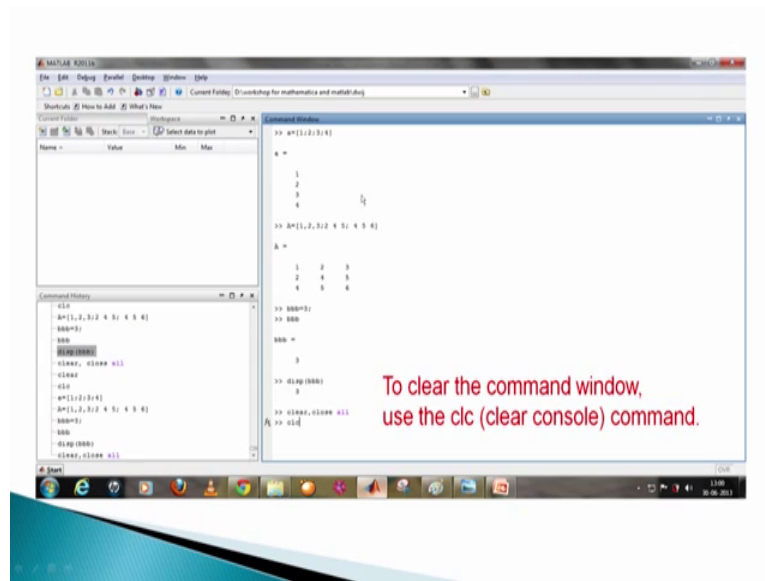
So, and for that this clear and close all command is very, very useful here.

(Refer Slide Time: 07:50)



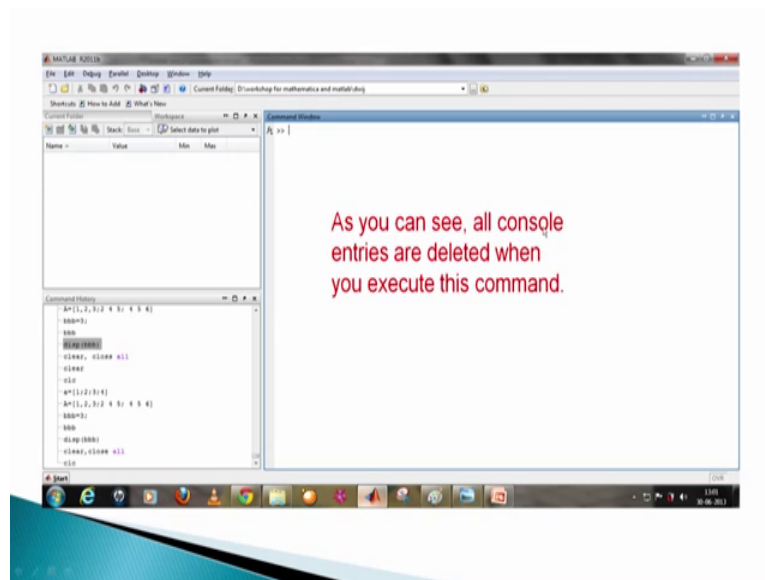
So, and when we use then you can see that all work space variables are deleted when you execute this command.

(Refer Slide Time: 07:56)



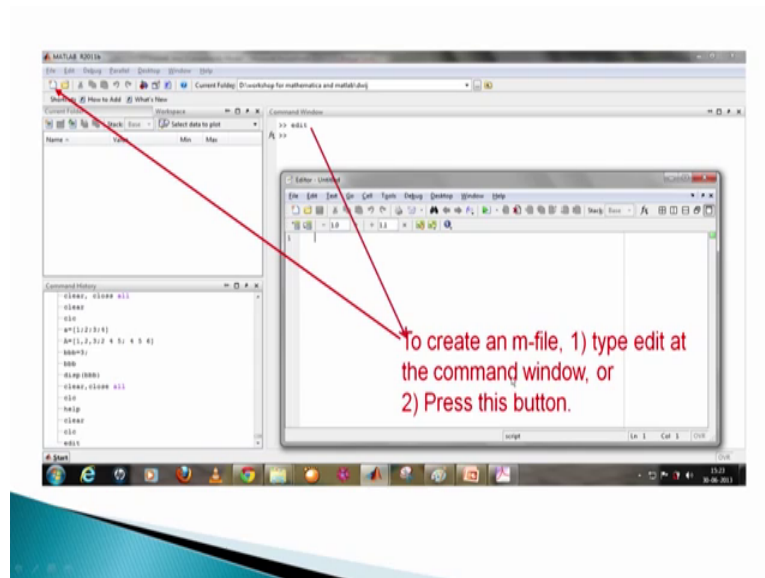
And to clear the command window this window you use `clc`. It is basically clear console, when you write `clc`, then all these entries are deleted.

(Refer Slide Time: 08:03)



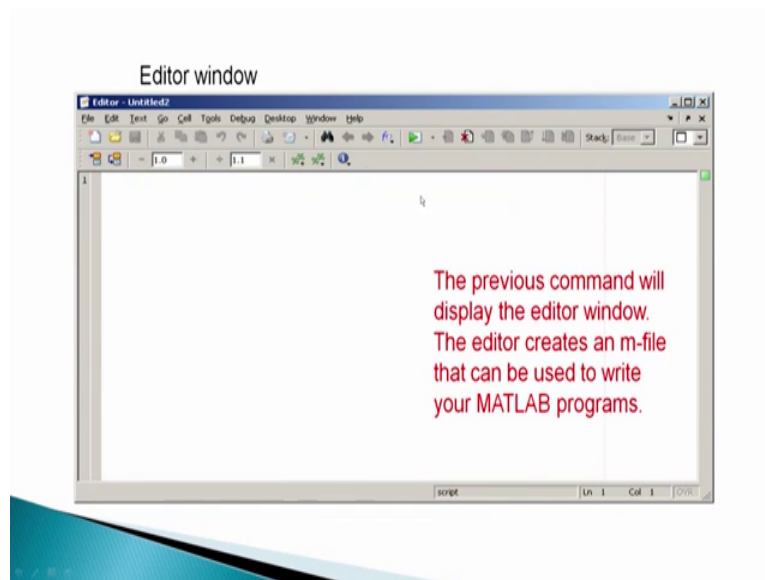
So, all console entries are deleted when you execute this command.

(Refer Slide Time: 08:11)



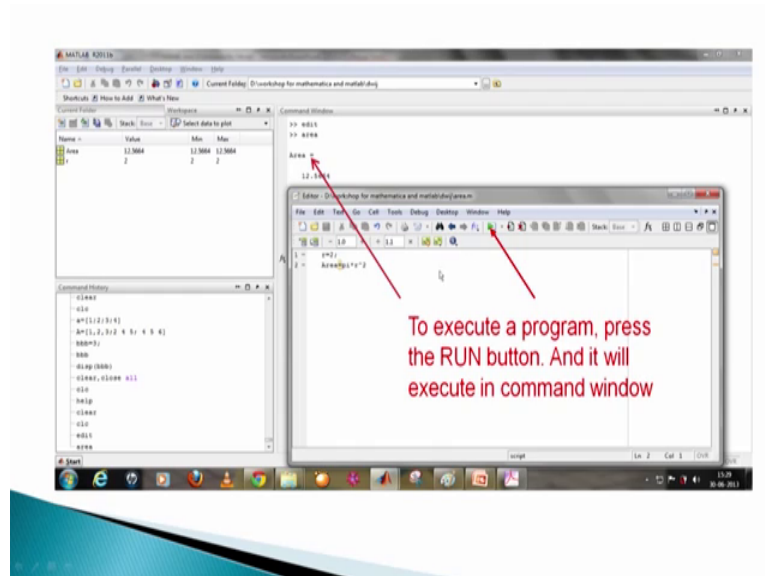
And to write down the MATLAB function or say m file, we can create m file by writing edit here, or by clicking this button. So, to create an m file type edit at the command window or to press this button, by pressing this you can have the editor window.

(Refer Slide Time: 08:30)



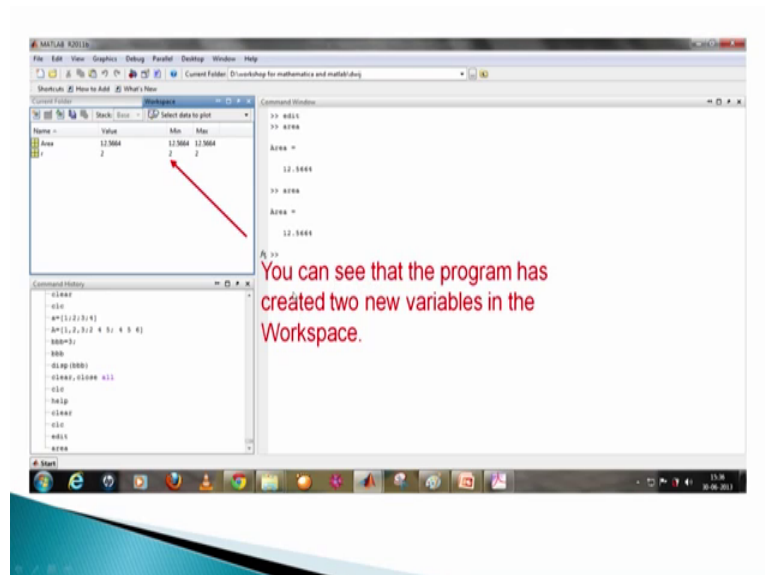
And here you can write your com say file, and you can create an m file that can be used to write your MATLAB programs.

(Refer Slide Time: 08:42)



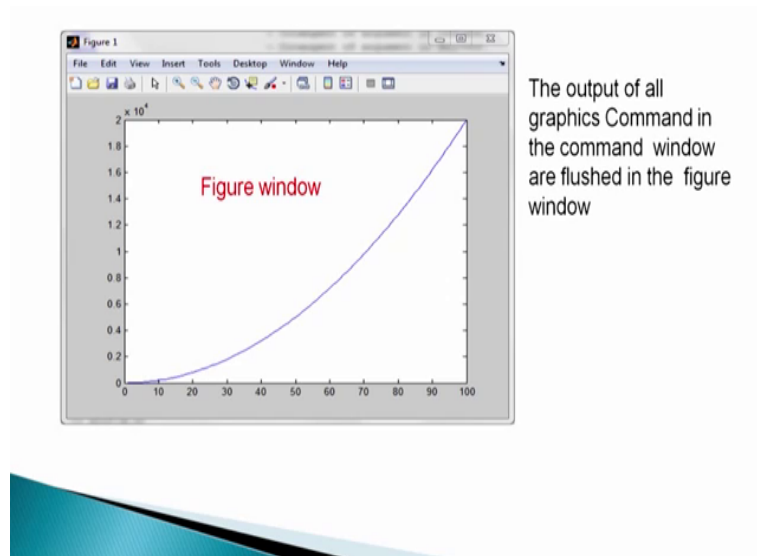
So, one such example is given here. You simply write this m file for finding the area of a circle. So, here we write R equal to 2, and area is given as pi R square. And to run this program there is a button here, and if you click this then this file will run. And it will execute in command window, and your output is given as areas 12.5664.

(Refer Slide Time: 09:15)



And by doing this now you can say that you can see that the program has created 2 new variables here, area is 12.5664, and minimum max is given here R is given as 2. So, and when you execute here then variable is created and it is listed in this file.

(Refer Slide Time: 09:32)



And the output of all graphics commands in the command window, are flushed in the figure window there is a figure window which is given here.

(Refer Slide Time: 09:42)

### How to get help:

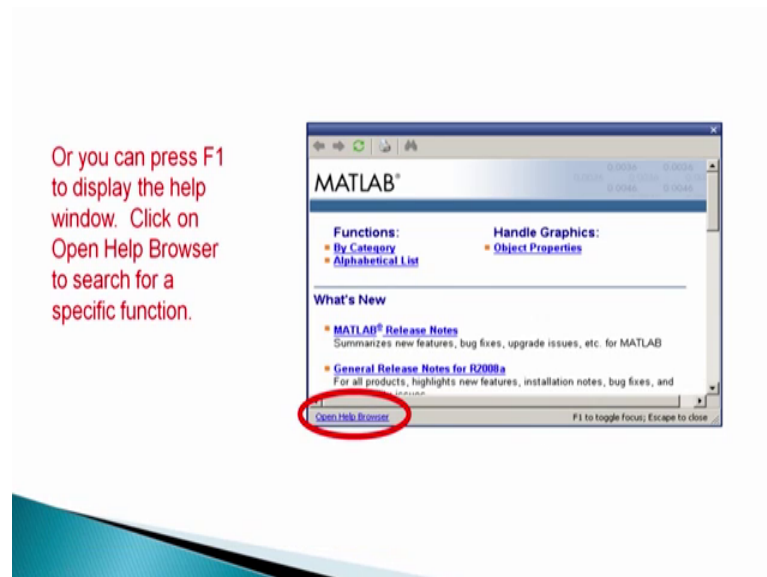
If you want to see help, you can type help at the command window.

For example if you want to find out how to use the tangent Function, just type >>help tan

And so, this is a basic environment in MATLAB. The most important thing in learning any kind of software is to find out how to get help. So, once we know how to get help, then you can explore yourself to find new commands and you can get all the information about the new command.

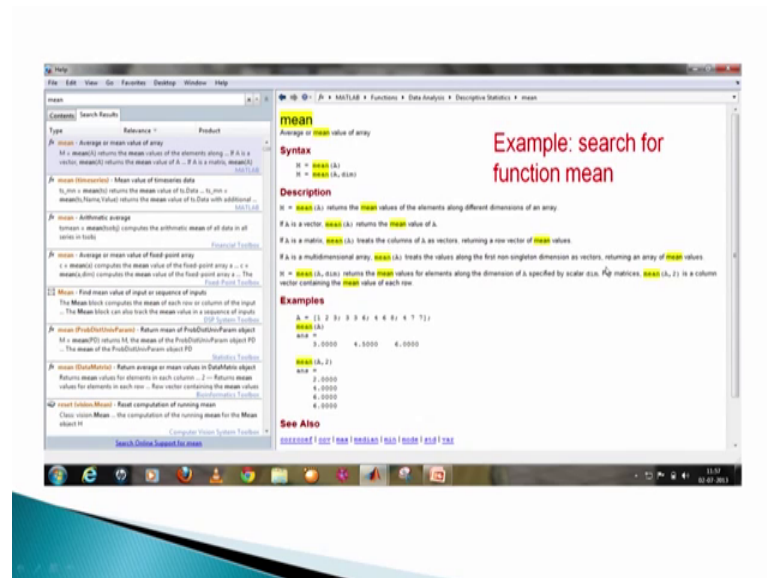
So, the important part of this lecture is how to get help. So, if you want to see help you can type help here, you can type help, and it will give you all the commands here. For example, if you want to find out how to use the tangent function just type help tan.

(Refer Slide Time: 10:23)



When you write help tan, you can say listed all the command which use tan. Or you can press f 1, your keyboard you press simply f 1. And it will display the help window. And click on the open help browser to search for a specific function. So, here you when you click this and then you just type whatever you want to type and it will list you all the topics related to that.

(Refer Slide Time: 10:53)



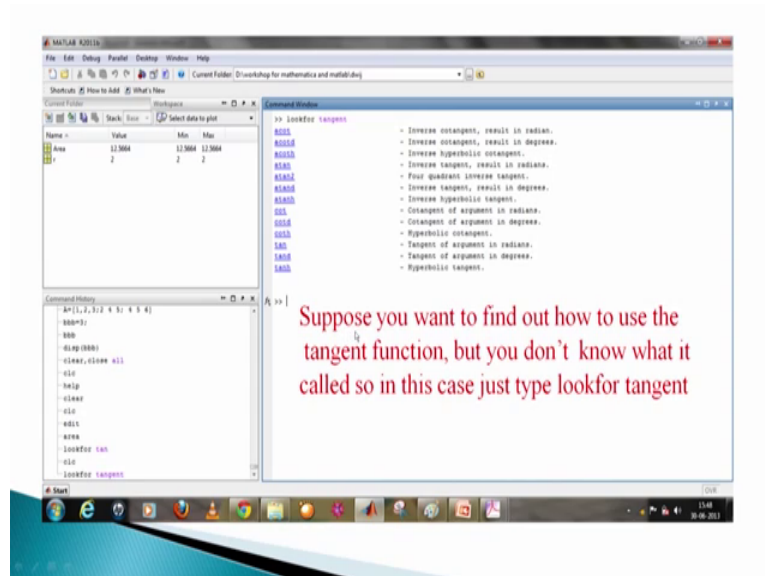
For example, if you want to search for mean function. Then you tie that as mean function here, and it will give you all this result here.

So, one such result is given here, mean is here you will get all the information about that here, this index is given how to use that mean, description is there example some examples are there how do you use this mean function. So, this is very, very useful when you are new to MATLAB, if you are expert in MATLAB, then maybe you can escape this.

But for bigness, it is always better to look for help, how to get help without say without bothering anywhere. So, rather than googling it is always better to use this command.



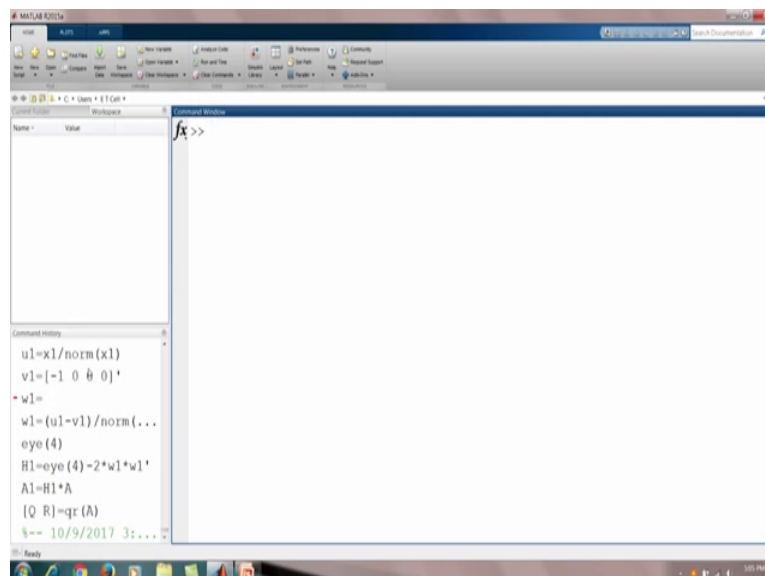
(Refer Slide Time: 11:47)



Now that and suppose you want to find out how to use a tangent function, but you do know the exact command how we can look at for tangent. So, in this case you can type look for tangent. So, these look for and whatever word you want to write it write here. And that will give you all the commands related to tangent. So, these are some command related to tangent, and this will list you here the explanation for these commands.

Now, let us consider the basic fundamentals. So, so far, we have just memorized with the windows available in MATLAB. Here it is the window which I am working here.

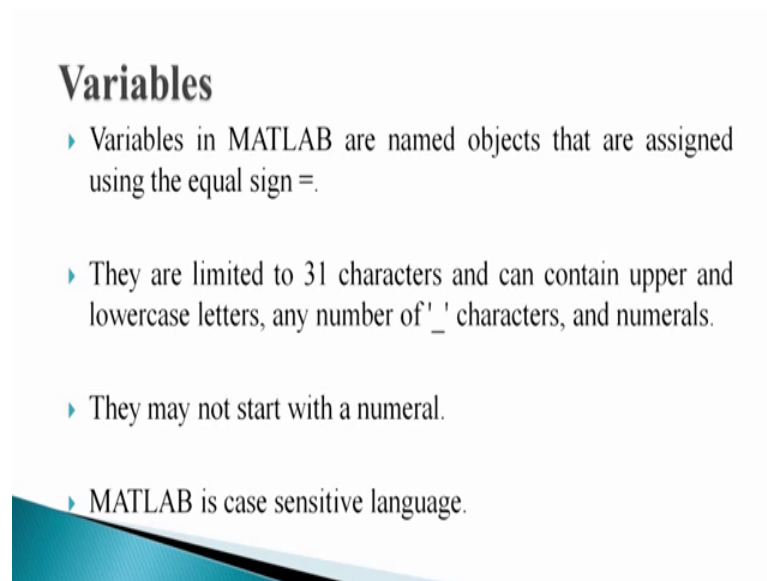
(Refer Slide Time: 12:32)



So, here this is a current directory which is in which we are working this is command window. And here it is the workspace, and here it is the command history. So, here we have used many things ok.

So, now let us go back to our presentation here. Now let us summarize with some basic fundamentals. So, first fundamental is the variables name, how we define variables.

(Refer Slide Time: 12:57)



## Variables

- ▶ Variables in MATLAB are named objects that are assigned using the equal sign =.
- ▶ They are limited to 31 characters and can contain upper and lowercase letters, any number of '\_' characters, and numerals.
- ▶ They may not start with a numeral.
- ▶ MATLAB is case sensitive language.

So, variable in MATLAB are named object that are assigned using the equal sign. So, this sign, we use this to define variables. And they are limited to 31 characters, and can contain upper and lower-case letters any number or this characters and numerals. They may not start with a numeral. So, please do not define any variable which start with numeral. MATLAB is a case sensitive language. So, small a and capital A have different mean.

(Refer Slide Time: 13:30)

## Variables

- ▶ **Example of some valid variables:**
  - ▶ a=1;
  - ▶ speed=200;
  - ▶ name='John Smith'
  - ▶ No\_of\_Students=100;
  - ▶ Id2=3208064;
- ▶ **Example of Invalid variable name :**
  - ▶ 2for1='yes';
  - ▶ first one=1;

And let us consider some example 4 variables.

So, a equal to 1, this will give you create a variable, whose value is 1. Speed is 200 name this a string john smith number of student number. And o underscore of underscore students is given as 100. I need to is given by this.

Ah then example of invalid variable name. So, here 2 for one it is started with numeral 1 so, this is not ok. First one there is a space here. So, that is not correct. So, these are invalid variable name, and these are with valid variable names.

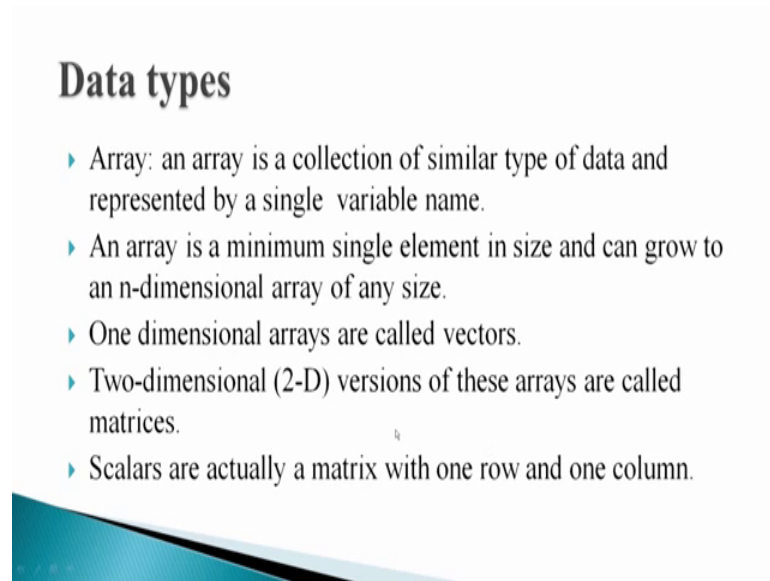
(Refer Slide Time: 14:14)

## Character set

- The character set is a set of characters that form a part of the statements written using the programming language.
- ▶ **Alphabets:** a-z and A-Z
- ▶ **Numerals:** 0-9
- ▶ **Special characters :** including escape.
- ▶ **White space characters:** tab, blank, new line, vertical tab and line-feed .

And character sets the character set is a set of characters that form a part of the statement written using the programming language. So, here we alphabets, small a to small z, and capital A to capital Z, numerals 0 to 9. Special characters including escaped whitespace character, tab, blank, new line, vertical line, vertical tab and line feed. All these are white space character.

(Refer Slide Time: 14:44)



### Data types

- ▶ Array: an array is a collection of similar type of data and represented by a single variable name.
- ▶ An array is a minimum single element in size and can grow to an n-dimensional array of any size.
- ▶ One dimensional arrays are called vectors.
- ▶ Two-dimensional (2-D) versions of these arrays are called matrices.
- ▶ Scalars are actually a matrix with one row and one column.

Now, data type so, here one array, and array is a collection of similar type of data, and represented by a single variable name. And array is a minimum single element size and can grow to an n dimensional array of any size. One dimensional array are called vectors. 2Dimensional version of these arrays are called matrices. And scalars are actually a matrix with one row and one column. In fact, here everything is denoted with the help of matrix.

So, if you write any word it is saved as one cross one matrix. So, that is why we are dealing with matrices. So, here are there are special constant and variables.

(Refer Slide Time: 15:31)

### Special constants and variables

S.no	Representation	value
1	Pi	3.14159
2	i	$\sqrt{-1}$
3	j	$\sqrt{-1}$
4	inf	$\infty$
5	Nan	Not a number
6	eps	Number small enough equivalent to zero
7	ans	Default output variable
8	nargin	No of input arguments of a function
9	nargout	No of output arguments of a function
9	Realmax	Largest real number

This pi which is having this value 3.14159, i and j represent the complex number under root of minus 1. Inf is infinity, nan for not a number, eps number small enough equivalent to 0. Ans that will simply give you default output variable, number of argument input. So, nargin, that is number of input argument of a function, number of argument out. So, number of output arguments of a function, real max largest real numbers.

(Refer Slide Time: 16:10)

## As a Calculator

After starting MATLAB, the command window with command prompt will appear

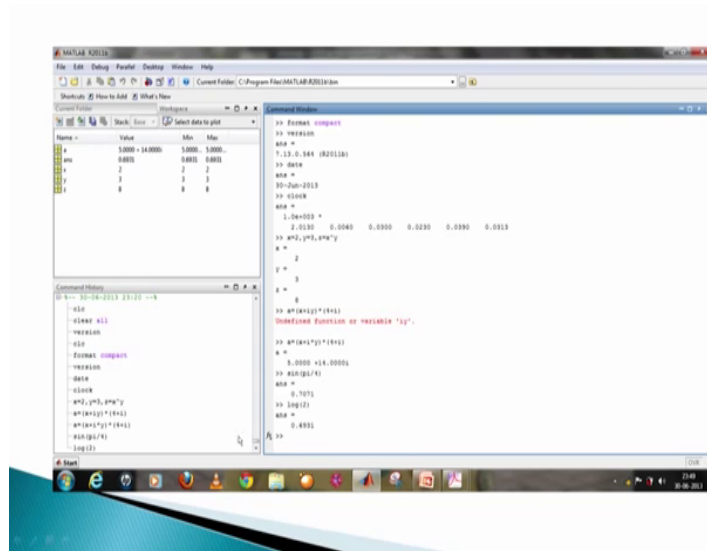
>>

Now we can do arithmetic operations(+,-,\*,/), exponentiation (^) and logarithms operations (log(x),log10(x)), trigonometric operations(sin(x),asin(x) and more) and complex operations involving complex numbers.

Note about: ans, format compact, format loose

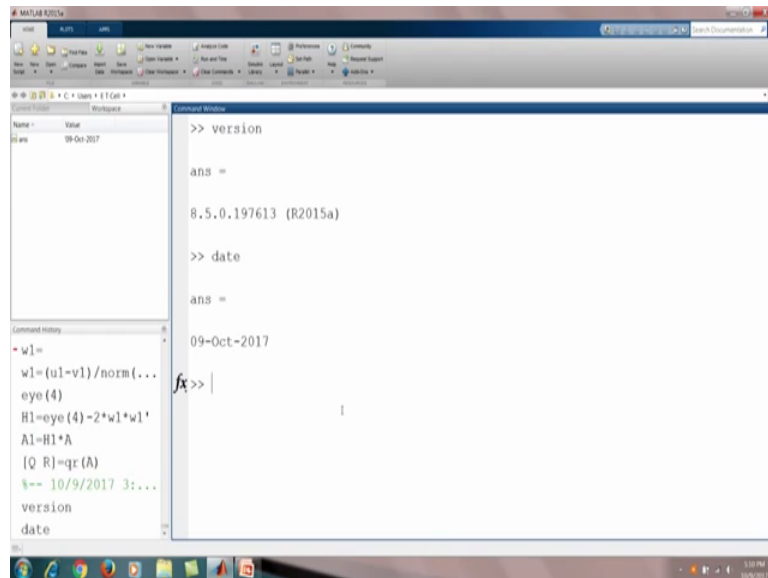
So, first let us try to consider the application of MATLAB as a calculator. So, after starting MATLAB the command window with command prompt will appear like this. Now we can do arithmetic operation whatever we want to write it here exponential this is exponential plus minus multiplication division and logarithmic operation such as  $\log x$   $\log_{10} x$  this log base ten  $\sin x$  trigonometric operations says that  $\sin x$  and many more. And complex operation involving complex numbers. And you can look at some answer format compact format loose that we will look at here.

(Refer Slide Time: 16:53)



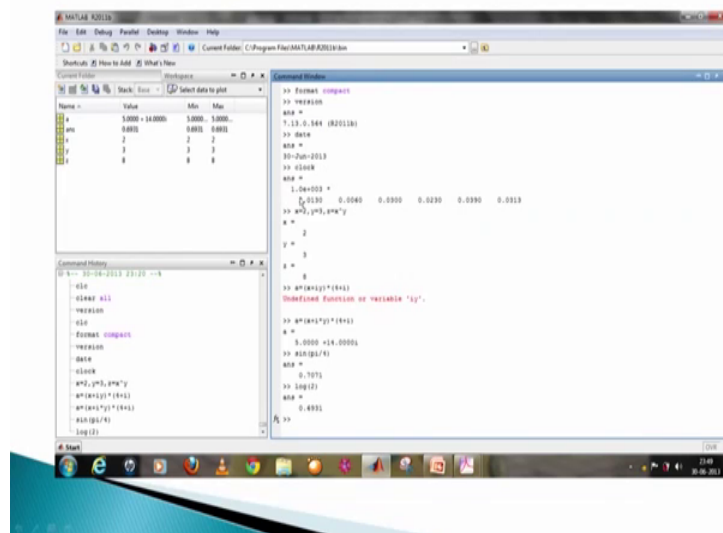
So, let us first look at this format compact. It means that we are looking very, and then we first command is version; which will give the information about the version of the MATLAB which you are using. So, when we write version as so, for this when we prepare this slide. At that time, I am using the version 7 version are 2011 be so, that is Lebanon and this slide is prepared as on the date. So, date is given here clock is and here are some calculation so, let us do this here.

(Refer Slide Time: 17:34)



So, let us have this here and then, when we write version then it will display the version of this current window where I am working. And then here we can write a date here, and date is 9th October 2017. So, by writing version it will give you the version of the current MATLAB version we are using, date it will give you the date of when I have prepare this slides.

(Refer Slide Time: 17:59)



And clock will give you the say timing time date everything of that particular name. And now let us use this as a calculator. So, x equal to 2 y equal to 3, and z let us say x square


x to power y. So, x is listed as 2, y is listed at 3, and z is 2 to power 3, and it is coming out to be 8. And similarly, we can do mini calculation so, for example, we want to define x plus i y cross 4 plus i and it will give you some results similarly sine pi I by 4. So, these are some command we here we are using MATLAB as a calculator.

(Refer Slide Time: 18:42)

## Output Format

Format Type	Result	Example
Short	Scaled fixed point format with 5 digits	3.1416
Long	Scaled fixed point format with 15 digits for double and 7 digits for single	3.141592653589793
Short e	Floating point format with 5 digits	3.1416e+000

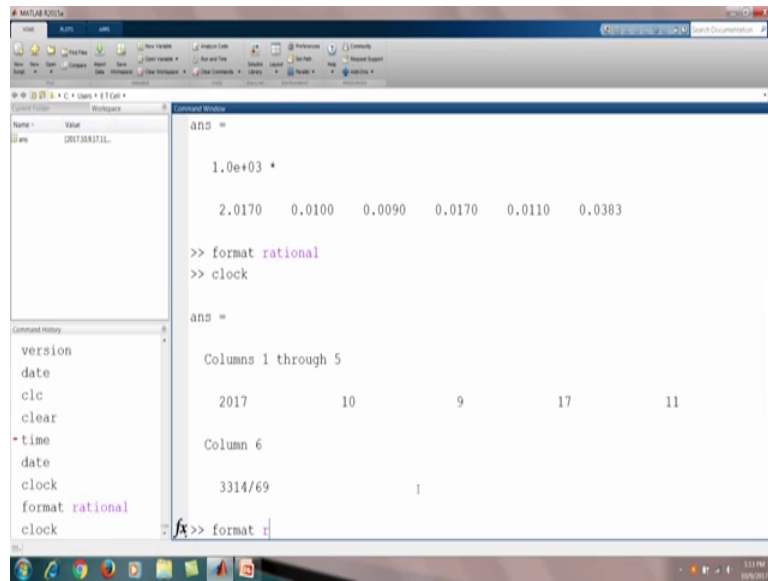
There are some more format, try them on your own. Some of them are as follows  
Long e, short g, long g, short eng, hex, rat, bank.



Now we have several output format. Say short that a scaled fixed-point format with 5 digit. For example, we can write 3.1416, that is a short form it, and long form it, that is scaled fixed point format with 15 digits for double and 7 digit for single. So, this is one example 3.14159 2 so, this is I am just writing the value of pi in format long and format shot. Similarly, we have short e that is floating point format with 5 digit and it is listed as this. So, here the value of pi in short e is given as this. There are many more format we can try those format on your own, and some of them are as follows long e, short g, long g, short in eng hex rat bank so, here for example, if I want to write say a format rational, right.



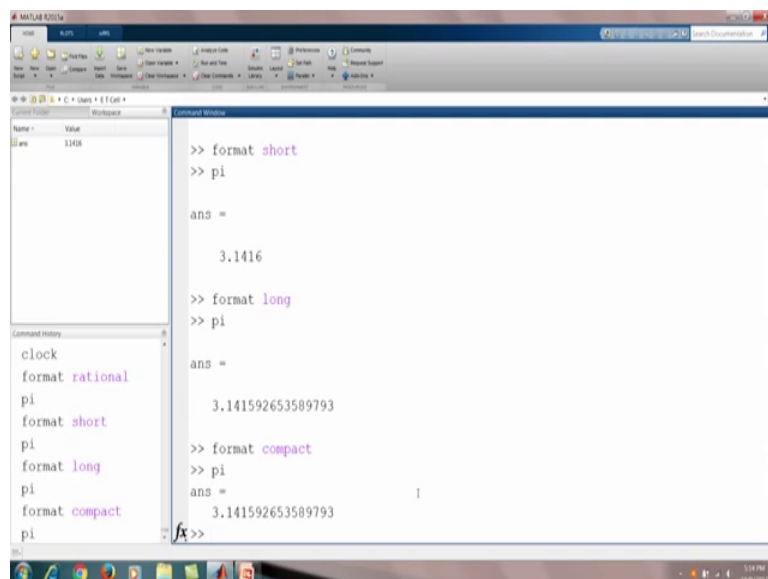
(Refer Slide Time: 19:35)



```
ans =  
  
1.0e+03 *  
  
2.0170 0.0100 0.0090 0.0170 0.0110 0.0383  
  
>> format rational  
>> clock  
  
ans =  
  
Columns 1 through 5  
  
2017      10      9      17      11  
  
Column 6  
  
3314/69      1  
  
fx>> format r
```

And then we can write pi, and the value of pi, is given as 355 divided by 113.

(Refer Slide Time: 19:45)



```
>> format short  
>> pi  
  
ans =  
  
3.1416  
  
>> format long  
>> pi  
  
ans =  
  
3.141592653589793  
  
>> format compact  
>> pi  
  
ans =  
  
3.141592653589793      1  
  
fx>>
```

And if you want to write format short, then you can write pi as 3.1416 that is format short then format long you can use. And then again you can write pi and it will have this value, and then format say compact, you can use then you can write pi as this value. And then we can use many more format, I am just listing only few very few. Then some of Useful commands are who and this who it will list all the variables in the current work workspace.

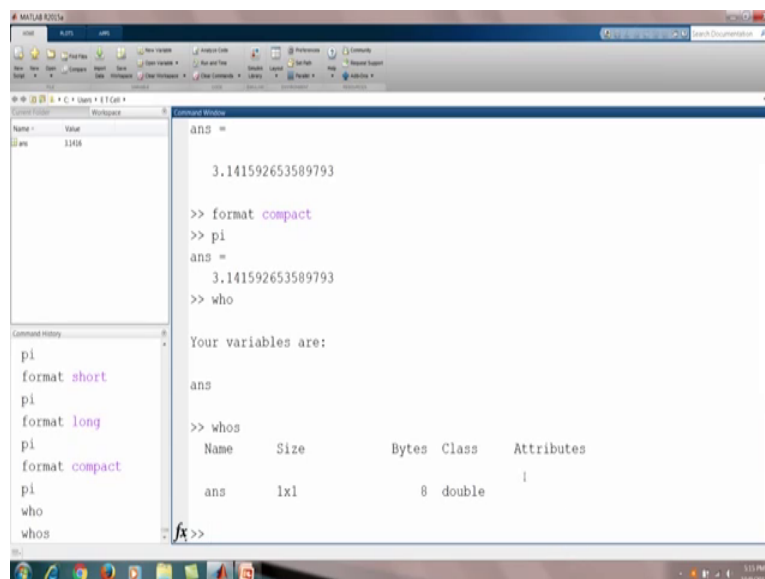
(Refer Slide Time: 20:25)

## Some of useful commands

- ❑ **who** :It lists the variables in the current workspace.
- ❑ **whos**: It lists all the variables in the current workspace, together with information about their size, bytes, class.
- ❑ **clear**: It Clears the variables and functions from memory.
- ❑ **clc** : It Clears the command window.

For example, in this workspace on we have only this variable.

(Refer Slide Time: 20:38)



So, if I write who it will give you that there only one variable answer. Now if you want to want to find out the size of this answer, then we write whose, then it will give you every information about the variable listed here. So, here we have only one variable that is size is one cross 1 by it is 8 bytes class is double, and if we have some kind of attribute then also it is listed here. So, whose it lists all the variables in the current workspace

together with the information about their size bytes. And class clear it clears the variables and function from memory clc it clears the command window that we have just shown.

(Refer Slide Time: 21:17)

- ▶ All of the data that you enter into MATLAB is stored in the form of a matrix or a multidimensional array. Even a single numeric value like 100 is stored as a matrix (in this case, a matrix having dimensions 1-by-1):

```
>> A = 100;
>> Whos A
Name Size Bytes Class
A 1x1 8 double array
```

Now, all of the data that you enter into MATLAB is stored in the form of a matrix or a multi diamond dimension array. Even a single numeric value like 100 is stored as a metric in this case matrix of the size one by one. So, if I write a as 100, and if we check the dimension of a then whose a is basically this name a size one cross 1 byte is 8, and class is double array.

(Refer Slide Time: 21:42)

## The Colon Operator

- ▶ To generate a vector of equally spaced elements MATLAB provides the colon operator.

▶ Ex:

```
>> a=1:5
```

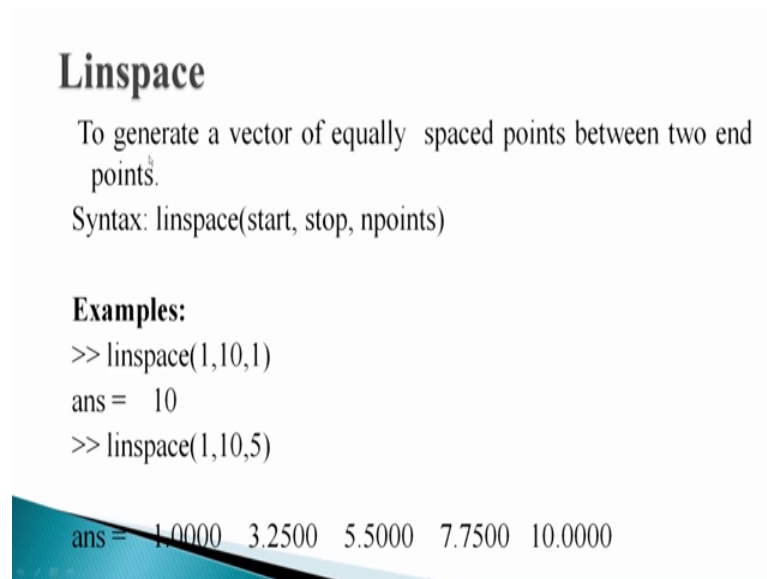
```
a = 1 2 3 4 5
```

```
>> b=0:2:10
```

```
b = 0 2 4 6 8 10
```

Now, let us consider some application of colon operator. The colon operator is used to generate a vector of equally spaced element element MATLAB provide the colon operator. So, here we have a equal to 1 colon 5. Then it will give you the equally spaced element between 1 to 5 so, that is 1 2 3 4 5. And when we do not write any we simply 1 to 5; means, from 1 to 5 we list all the element. And when we write 0 colon 2 colon 10. So, it means that between 0 to 10, we need space of 2. So, this is initial point, this is end point, and this is the space 2, right so, it will give you 0 2 4 6 8 10.

(Refer Slide Time: 22:34)



### Linspace

To generate a vector of equally spaced points between two end points.

Syntax: `linspace(start, stop, npoints)`

**Examples:**

```
>> linspace(1,10,1)
ans = 10
>> linspace(1,10,5)
ans = 1.0000 3.2500 5.5000 7.7500 10.0000
```

Line space to generate a vector of equally spaced points between 2 end points that is we can write line space start point, stop point and number of points. So, for example, if I write line space from 1 to 10, and we have only one point we want then it will give you nothing but 10. And line space from starting from 1 to 10, and we want number of point that is 5, then we have these many points listed here.

So, that will generate a vector of equally spaced points between 2 end points.

(Refer Slide Time: 23:06)


## logspace

Logspace(a,b,n): creates a logarithmically spaced vector of length n in the interval  $10^a$  to  $10^b$

Example:

```
> logspace(0,4,3)
```

ans =




Then log space creates a logarithmically spaced vector of length n in the interval  $10^a$  to  $10^b$ . So, as a starting point a is the end point. So, for example, logspace 0 4 3, this is a, this is b, and this 3 is the number of points between 0 and 4. So, here we have  $10^0$ ,  $10^2$  and  $10^4$ . So, basically, we have we want 3 points, 1, 100, and 10000 is given here.

(Refer Slide Time: 23:40)

## Matrix Variables

- ▶ Matrix variables are initialized similar to single variables.
- ▶ The values in a matrix variable is defined in square brackets.
- ▶ Example:
  - `aaa = [1,2,3,4];`
  - `bbb = [1;2;3;4];`



Now matrix variable matrix variable are initialized similar to single variable, and the values of a matrix variable is defined in a square matrix. So, when we define any matrix

then it is given in terms of square. So, it triple a is 1 2 3 4, and this semi column just to prevent the display otherwise when you do not put this semicolon, then it will display every time we define a variable. And if you to avoid that you can put this semi column and it will not display the result.

So, triple a is 1 2 3 4 and triple b is 1, colon 2, colon semicolon to semicolon 3 semicolon 4.

(Refer Slide Time: 24:21)

## Matrices

- ▶ To create a 3x3 matrix A that has 1,2,3 in first row, 2,3,4 in the second row and 5 7 3 in the third row:  
`>>A=[1 2 3;2 3 4; 5 7 3]`

The semicolon is used here to separate rows in the matrix.  
MATLAB gives you

A =

1	2	3
2	3	4
5	7	3

If you do not want to write you can use simply do not put any kind of colon or semicolon here. To create A 3cross 3 matrix A that has 1 2 3 in first row, 2 3 4 in the second row, and 5 7 3 in the third row, you can use this command. A equal to within bracket is big bracket 1 2 3, we are put in using a space here. We are not putting comma here. And then semicolon and when you put semicolon it means that first row is finished or the particular row is finished. 2 3 4 and colon which means that second row is also finished 5 7 3 and so on. The semicolon is used here to separate rows in the matrix. And MATLAB will give you 1 2 3 2 3 4 and 5 7 3 here.

(Refer Slide Time: 25:20)

- ▶ If you don't want MATLAB to display the output of your command, put a semicolon at the end of the command

```
A=[1 2 3;2 3 4; 5 7 3];
```

Matrix A has been created but MATLAB does not display the result.

- ▶ Semicolon is necessary if you are doing a lengthy calculation and don't want to see everything on the screen.

So, in this way you can use MATLAB to find out your matrix. And if you do not want MATLAB to display the output of your command, put a semicolon at the end of the command. So, if you put this semicolon it will not display your command is it ok. So, matrix a has been created, but MATLAB does not display the result. And if you look at here, I am not this put this semicolon. So, it will display our result, but if we put a semicolon, then will not display this it will store, but not displayed when you require you can display that thing. So, semicolon is necessary if you are doing a lengthy calculation and do not want to see everything on the screen. So, we want that it is stored, but you not to display each and everything.

(Refer Slide Time: 25:55)

```
>> A =
     1     2     3
     2     3     4
     5     7     3
>> A(2,3)
ans = 4
>> A(3,:)
ans =
     5     7     3
>> A(:,2)
ans =
     2
     3
>> A(3,1:2) % 1st and 2nd element in 3rd row
ans =
     5     7
>> A.*A % element wise multiplication
ans =
     1     4     9
     4     9    16
    25    49     9
>> A*A % Matrix multiplication
ans =
    20    29    20
    28    41    30
    34    52    52
```

So, for example, if a is 1, 2, 3, 2, 3, 4, 5, 7, 3 now, we want to work with say, these structure of this matrix same. So, if I look at A 2 3 so, that will give you the second row and this third. So, here it is what second row, and the third element. So, A 2 3 is this. So, A 2 3 is simply the 2 3 element of a matrix a. So, that is nothing but for now a 3Dash colon. So, it will give you all the entries of the third row

So, A 3colon means 5 7 3 so, A 3comma colon. So, in third row all the element now a colon comma 2 so, it will give you all the element in second column, that is 2 3 7. So, it is listed here. Now here when we write A 3 comma 1 3 comma 1 colon 2, it will give you in third row one 2 2 element of third row, first and second element in third row.

So, that will give you 5 and 7. Now a dot star a that is element wise multiplication. So, this will give you 1 1, A 1 1 into b 1 1, A 2 2, b 2 2 and so on. So, that will give you element wise multiplication. But when we want matrix multiplication will not use this dot here. We simply write A cross A. That will give you matrix multiplication. So, A square is given by this, and this is not a square that is my element wise multiplication. So, please see the difference between this a dot square cross star A or A star A, right?



(Refer Slide Time: 27:47)

```
Command Window
>> A=[1 2 3;2 3 4; 5 7 3];
>> A^2
ans =
    20    29    20
    28    41    30
    34    52    52
>> A.^2
ans =
     1     4     9
     4     9    16
    25    49     9
>> A/pi
ans =
    0.3183    0.6366    0.9549
    0.6366    0.9549    1.2732
    1.5915    2.2282    0.9549
>> A*[1 2 3]
a =
     1     2     3
>> a*A
ans =
    20    29    20
>> I=eye(3)
I =
     1     0     0
     0     1     0
     0     0     1
>> B=zeros(2,3)
B =
     0     0     0
     0     0     0
fx >> |
```

So, here we have just listed one example. A is given as 1 2 3 3 2 3 4 5 7 3. And a square if you want to find out then it is it will give you A into A. So, that is given here, but if a dot a square, then it is element wise square. So, that is 1 4 9 4 9 6. So, please try to look at here that A dot square, and A square is having different meaning. Here A square is simple A into A and it is that element to power 2, is it ok? A divided by pi is given by this. A you define A as 1 2 3, then we can define matrix multiplication. And this is the symbol I size 3 this will give you the identity matrix of the size 3 cross 3. And this 0es is basically 2 cross 3 matrix of all the element 0. So, 0es and I is a special command which will give you the identity matrix or a 0 matrix. So, if you want to access single element then how we can do.

(Refer Slide Time: 28:54)

### Accessing Single Elements

- ▶ A(row, column)

```
>> A = magic(4)
A = 16 2 3 13
     5 11 10 8
     9 7 6 12
     4 14 15 1
```

you would access the element at row 4, column 2 with

```
>> A(4, 2)
ans = 14
```

So, first says like let us create a matrix of size 4 so, A magic 4. So, that will give you a special matrix, 4 cross 4 matrix. Rows if you find out the row sum or the column sum or any and they it will give you a fixed value.

So, for once we have a matrix, then we can access our element 4 comma 2 that will give you 4th row comma and second element that we have just listed A 4 2 is nothing but 14 that is 4th row and second element.

(Refer Slide Time: 29:34)

### Specifying All Elements of a Row or Column

- ▶ The colon by itself refers to *all* the elements in a row or column of a matrix.

```
a = magic(3)
a =
     8     1     6
     3     5     7
     4     9     2
```

>> a(:,1) % all rows and first Column.

```
ans =
     8
     3
     4
```

The colon by itself refers to all the elements in a row or column of a matrix that we have seen. That A colon comma one, that gives you all rows and first column. So, all rows and first column that is all rows and first column.

(Refer Slide Time: 29:48)

```

>> a=[1 2 3; 4 5 6; 7 8 9]
a = 1 2 3
    4 5 6
    7 8 9
>> a(3,2)
ans = 8
>> a(2,3)
ans = 6
    9
>> a(2,:)
ans = 4 5 6
>> a(:,3)
ans = 3
    6
    9
>> a(4)
ans = 2
>> a(8)
ans = 6

```

Similarly, A 32 we know A 2 colon 3 comma 3, in third column, second and third element that is 6 9. So, third column second and third element, and this a 2 comma colon basically what? Second row all the element, second row all the element A colon comma 3. So, third column all the entries, third column all the entries A 4 now, when you write A 4 or 8, then it will use some other the A 4 is it is listed as a long one is single vector. Then a 2 is A 4 is basically what? A 4 is 1 4 7 and 2; so, this is your 4th entry.

A 8 is basically what? 3 6 and then 7 8 so, 6 value right. So, when you write A 4 or A 8, then it will list as column and next column and next column.

(Refer Slide Time: 30:52)

```
>> a(:)

ans =

     1
     4
     7
     2
     5
     8
     3
     6
     9
```

So, you can write a all colon, the a colon that will list, your matrix a as a column vector, right? And here you can find out what is A 4 or A 8 values, is it?

(Refer Slide Time: 31:06)

```
Deleting Rows or Columns
4

>> a=[1 2 3; 4 5 6; 7 8 9]

a=  1  2  3
    4  5  6
    7  8  9

>> a(:,2)=[];
>> a

a=  1  3
    4  6
    7  9
```

And deleting rows or columns if you want to delete something, then also we can use we can do in MATLAB. In fact, we have a matrix like A, then a colon comma 2 is blank. So, it means that this second column, all rows and second column is deleted. So, when we list now A, then the second 2 5 8 is gone. And in you know what is new A is 1 4 7 and 3

6 9. So, by this you can delete a particular column, if when you want to delete a row, then a that particular row which we want to delete comma that all colon.

(Refer Slide Time: 31:49)

### Some More Matrix operations

- ▶ Try out your self

```
>>A=[1 2 ; 7 4];  
>>B=[2 3; 1 1];  
>>a=[1 -1];  
>>b=[-1 1];
```

Then find the following

- ▶ A+B
- ▶ a+b
- ▶ A-B
- ▶ a-b

So, that will delete that thing. Now we will wait something and you may try on your own. So, you list one a matrix b a. Small a small b, then you can use the following operations. I suggest you try it on your own A plus B small a plus b A minus B a minus small a minus b.

(Refer Slide Time: 32:10)

- ▶ A'
- ▶ a'
- ▶ A\*B
- ▶ A.\*B
- ▶ A\b(left division)
- ▶ A\B
- ▶ A/B(Right division)
- ▶ inv(A)
- ▶ sum(a)
- ▶ sum(A)
- ▶ rank(A)
- ▶ trace(A)
- ▶ 2\*A
- ▶ A-2
- ▶ A.^3
- ▶ mean(A)
- ▶ max(A)
- ▶ min(A)
- ▶ sort(A)

All these operation you try on your own. A dash small a dash, A cross B, A dash A dot cross B, all this you try to find out sum a sum capital a, rank a trace a 2 cross a all this you please try and some more matrix operation which is relevant to this particular course.

(Refer Slide Time: 32:25)

## Some More Matrix operations

- ▶ `rref(A)`: row reduced echelon form of A
- ▶ `norm(A, '-')`: norm of a matrix or vector. Some common norms are `norm(A, 2)`, `norm(A, 1)`, `norm(A, 'inf')`, `norm(A, 'fro')`, `norm(A, 'inf')`.
- ▶ `cond(A, '-')`: condition number of a matrix.
- ▶ `[U S V]=svd(A)`: Singular value decomposition of a matrix A.
- ▶ `[V D]=eig(A)`: eigenpair of A.
- ▶ `pinv(A)`: pseudo inverse of A.
- ▶ `x=pinv(A)*b`: least square solution of  $Ax=b$ .
- ▶ `[Q R]=qr(A)`: QR decomposition of a matrix A.
- ▶ `[Q R]=schur(A)`, Schur form of a matrix A.

So, first is `rref A` that is row reduced equivalent form of the matrix A. So, this is the command for finding the row reduce equivalent form of a matrix A. Norm of A dash here you can put say command norm in which you want to find out the norm of a matrix. For example, if we write `norm A comma 2`, all these are small this is not a capital one. Please all these commands are a small command. So, `norm A comma 2` it will give you the matrix norm of A to matrix norm of m norm A. Comma 1 1 matrix norm of a norm a dash inf that will give you infinity norm of A for being norm of A that is given a norm a comma dash. So, and this is infinity norm.

Condition number `cond`, `cond A dash` this the here you will display the in which norm you want to find out the condition number. For example, 2 norm then you have to put 2 here, and if it is one norm you put one norm and so on. So, `cond A dash` this will give you condition number of matrix. Now if you do not use this only you write `norm of A` then it will give you the norm or the condition number only with respect 2 norm. So now, the singular value decomposition of a matrix A is given by `U S V`, that is `S V D` of a this is the command, and the output is given an in the form of 3 matrices, it is given as U. So,

U S V, S V D A will give you the singular value decomposition of a matrix A. And the eigenpair of a can be written as V D is equal to eigen of A. So, this will calculate the eigenpairs. And we represent the i matrix of eigenvectors, and d represent the eigenvalue of the matrix A. P in V A that will give you pseudo inverse of a matrix A.

So, once we have a matrix A by this p in way of A that give you pseudo inverse of A. And if we want to find out the inverse you remove this p, and we write I n V A that will give you inverse of the matrix A if a size of a is say square matrix. Then least square solution of A x equal to b in it is given as x equal to p inverse of a cross b. So, that will give you the least square solution of A x equal to b. These are some command which we are going to use it later on.

So, as this course is run you will see that these commands are very, very useful. And then Q R, that is Q R A that will give you the Q R decomposition of a matrix A. And the output is 2 matrices Q and R, where Q is orthogonal matrix. And R is your upper triangular matrix, and then there is one more command s c h u r A that will give schur form of matrix A. And again, the output is to Q and R where Q is orthogonal matrix and R is a upper triangular matrix.

So, that will give you the schur form of a matrix A.

(Refer Slide Time: 35:58)

## Some special Matrices

- ▶ `eye(m,n)` returns an mxn matrix with ones on the main diagonal
- ▶ `zeros(m,n)` mxn matrix of zeros
- ▶ `ones(m,n)`
- ▶ `rand(m,n)` mxn matrix of random numbers
- ▶ `magic(N)` is an N-by-N matrix constructed from the integers 1 through  $N^2$  with equal row, column, and diagonal sums.
- ▶ `diag(v)` generates a diagonal matrix with v in the diagonal

And there are some special matrices eye  $m$  cross  $n$  that gives you the identity matrix of size  $m$  cross  $n$ , where we have only one on the main diagonal rest are all 0s.  $0s$   $m$  cross  $n$  that will give you the matrix of size  $m$  cross  $n$ , and every element is 0, once  $m$  cross  $n$ . So,  $m$  cross  $n$  matrix with every entry one-rand  $m$   $n$ . That is  $m$   $n$  cross matrix of random numbers, magic  $n$  is an  $n$  by  $n$  matrix constructed from the integers one through  $n$  square, with equal row column and diagonal sum. So, that we have that you can verify diagonal of  $v$  generates a diagonal matrix with  $v$  in diagonal. And if we write diagonal of  $a$  that it will give you the diagonal element of a given matrix  $a$ .

Determinant of  $A$  that will give you the determinant of a matrix rank of  $a$  that will give you.

(Refer Slide Time: 36:54)

### Some useful commands related to matrices

- ❑ **det(A):** It returns the determinant of a given square matrix  $A$ .
- ❑ **rank(A):** It returns the rank of a given rectangular matrix  $A$ .
- ❑ **trace(A):** It returns the sum of the diagonal elements of a rectangular matrix  $A$ .
- ❑ **Inv(A):** It returns the inverse of a non-singular matrix  $A$ .
- ❑ **eig(A):** It returns the eigen values of matrix  $A$ .
- ❑ **[x, v]=eig(A):** It returns the eigen vectors  $v$  and eigen

The rank of the given matrix trace, inverse eigenvalue and when we write  $x$   $y$  as eigenvalue of  $a$  it will give you the eigenvectors as well as the eigenvalue. sorty sort each column of the matrix  $e$  in the ascending order.



(Refer Slide Time: 37:11)

```
□ Sort(E): sort each column of the matrix E in the ascending order.

>> a=[ 2 6 -3;8 4 10; 4 2 4];
>> a
a =
     2     6    -3
     8     4    10
     4     2     4

>> sort(a)
ans =
     2     2    -3
     4     4     4
     8     6    10
```

So, when we write a as this, then a can be sorted as this a is written as this and sort a means every column is sorted in ascending order. So, if you look at 2 8 4 is the first column. And when you apply sort then it is started as 2 4 8 6 4 2, it is sorted at 2 4 6 and minus 3 10 4 it is minus 3 4 10.

So, it is given in this way so, that is sorting. So, here we have seen certain common command related to matrices. And how to open MATLAB, and how to get help and how we can use some common commonly used command related to matrices. There are many more which we can consider, but that you can use help window and they are tutorial available on MATLAB, you can go and memorise yourself with MATLAB.

So, with this we end our lecture and we will continue in next lecture, thank you very much for listening us.

Thank you.