


MATLAB Course
For Engineers
lecture 1


MATLAB stands for MATrix LABoratory.

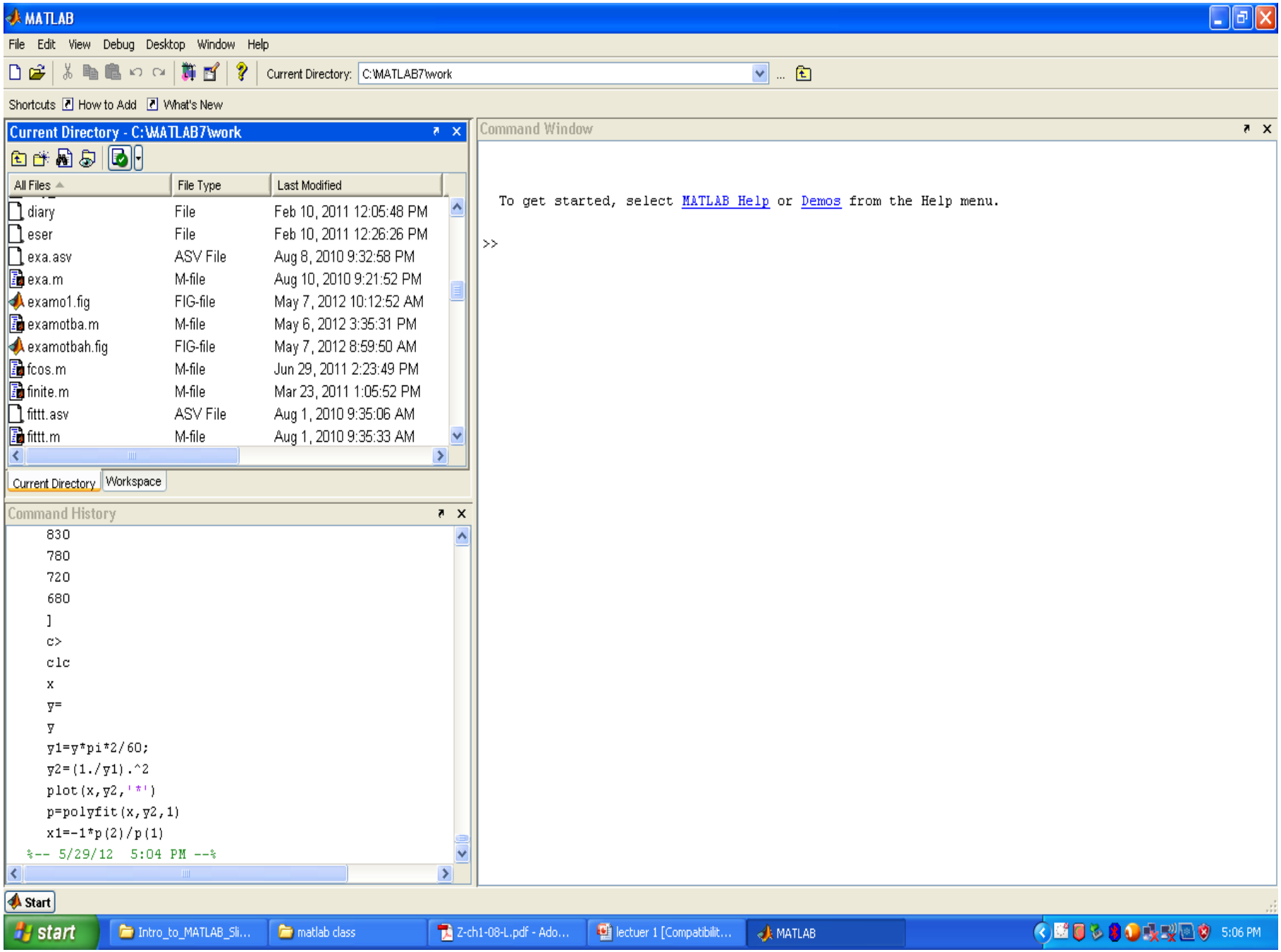
➢What is MATLAB?

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning.

`

Matlab Interface

1. **Command window:- Main Window, Enter variables, run programs.**
2. **Current directory window:- Shows the files in the current directory.**
3. **Workspace window:- Provides information about the variables that are used.**
4. **Command History window:- Logs commands entered in the command window.**
5. **Launch pad window:- Provides access to tools, demos and documentation.**
6. **Help window:- Provides help information.**


MATLAB
File Edit View Debug Desktop Window Help
Current Directory: C:\MATLAB7\work
Shortcuts How to Add What's New
Current Directory - C:\MATLAB7\work
All Files File Type Last Modified
diary File Feb 10, 2011 12:05:48 PM
eser File Feb 10, 2011 12:26:26 PM
exa.asv ASV File Aug 8, 2010 9:32:58 PM
exa.m M-file Aug 10, 2010 9:21:52 PM
examo1.fig FIG-file May 7, 2012 10:12:52 AM
examotba.m M-file May 6, 2012 3:35:31 PM
examotbah.fig FIG-file May 7, 2012 8:59:50 AM
fcos.m M-file Jun 29, 2011 2:23:49 PM
finite.m M-file Mar 23, 2011 1:05:52 PM
fittt.asv ASV File Aug 1, 2010 9:35:06 AM
fittt.m M-file Aug 1, 2010 9:35:33 AM
Current Directory
Workspace
Command History
830
780
720
680
]
c>
clc
x
y=
y
y1=y*pi*2/60;
y2=(1./y1).^2
plot(x,y2,'*')
p=polyfit(x,y2,1)
x1=-1*p(2)/p(1)
%-- 5/29/12 5:04 PM --%
Command Window
To get started, select MATLAB Help or Demos from the Help menu.
>>
Start
start
Intro_to_MATLAB_Sli...
matlab class
Z-ch1-08-L.pdf - Ado...
lecture 1 [Compatibilit...
MATLAB
5:06 PM

Using Matlab as a Calculator

- **Matlab Operations**

- **Addition +**
- **Subtraction -**
- **Multiplication ***
- **Right division /**
- **Left division **
- **Power (Exponentiation) ^**

- **Order of Precedence**

1. **Parentheses ( )**
2. **Power ^**
3. **Multiplication and division with equal precedence, evaluated from left to right.**
4. **Addition and subtraction with equal precedence, evaluated from left to right.**

Some commonly used mathematical functions

Function	Matlab syntax	Function	Matlab syntax
e^x	exp (x)	$\|x\|$	abs(x)
$\sqrt{x}$	sqrt (x)	$x!$	factorial (x) x>0
$\ln x$	log(x)	π	pi
$\log_{10} x$	log10(x)	∞	inf
$\cos x$	cos(x) *	$\sin^{-1}(x)$	asin(x)
$\sin x$	sin(x)	$\cos^{-1}(x)$	acos(x)
$\tan x$	tan(x)	$\tan^{-1}(x)$	atan(x)
$\cot x$	cot(x)		
$\sec x$	sec(x)		
$\csc x$	csc(x)		

* The Matlab trigonometric functions use radian measure

An Example :

```
>> 8/10
ans = 0.8000

>> 100*(0.5+1)
ans = 150

>> sin(pi/2)
ans = 1

>> tan(deg2rad(45))
ans = 1.0000

>> atand(1)
ans = 45

>> sin(pi/4)^2
ans = 0.5000
```

Problems:-

$$\cos\left(\frac{5\pi}{6}\right).\sin\left(\frac{7\pi}{8}\right)+\frac{\tan\left(\frac{\pi}{6}\ln 8\right)}{\sqrt{7}+2}$$

$$23\left(-8+\frac{\sqrt{40}}{5}\right)+\left(\frac{\ln 70}{8^2}+\log_{10}4.7\right)^2$$

Ans = 0.2846
Ans = -154.3617

Variables

* Rules a bout variable Names

1. Must begin with a letter.
2. Case sensitive , A not equal a.
3. Can be up to 63 characters.
4. Can contain letters, digits, and underscore.
5. Can't contain punctuation characters (e.g comma, semicolon ;)
6. Avoid using the names of a built- function.
7. No space.

- The Assignment Operator (=) is used to assigns a values to a variable
- Typing x=3 assign the value 3 to the variable x.
- We can then type x=x+2. This assigns the value 3+2=5 to x.
- In algebra we can write x+2=20, but in Matlab we cannot ?????.
- In Matlab the left side of the = operator must be a single variable.
- The right side must be a computable value.

Display Formats

1. Format short: Fixed- point with 4 decimal digits e.g 290/7 ans = 41.4286
2. Format long : Fixed- point with 14 decimal digits e.g 290/7

   ans = 41. 42857142857143
3. Format short e : Scientific notation with 4 decimal digits e.g 290/7

   ans = 4.1429e+001
4. Format long e: Scientific notation with 16 digits (15 decimal) plus exponent

   e.g 290/7 ans = 4.142857142857143e+001
5. Format short g : Best of 5 digits fixed e.g 290/7 ans = 41.429
6. Format long g : Best of 15 digits fixed e.g 290/7 ans = 41.4285714285714
7. Format bank : Two decimal digits e.g 290/7 ans = 41.43
8. Format rat : Rational approximation e.g 43/7 ans = 43/7

Rounding Function

1. round (x): Round to the nearest integer e.g round(17/5) or round(3.4)

   ans = 3
2. fix(x) : Round towards zero e.g fix(13/5) or fix(2.6) ans = 2
3. ceil(x) : Round towards infinity e.g ceil(11/5) or ceil(2.2) ans = 3
4. floor(x) : Round towards minus infinity e.g floor(-9/4) or floor(-2.25)

   ans = -3
5. rem (x,y) : Returns the remainder after *x* is divided by *y* rem(13,5) ans = 3
6. mod (x,y) : Returns the remainder after *x* is divided by *y* mod(13,5) ans =3

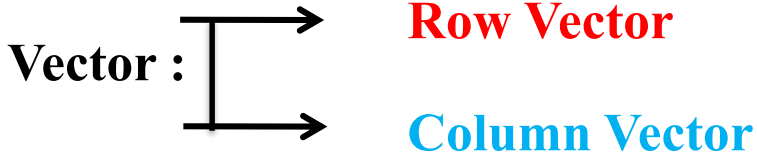
Note : rem and mod is the same for positive sign.

For negative sign mod(-x,y)=rem(-x,y)+y

Useful Commands for Managing Variables

1. clc : Clears the command window.
2. clear : Removes all variables from memory.
3. clear var1,var2 : Removes the variables var1 and var2 from memory.
4. who : Lists the variables currently in memory.
5. whos : Lists the current variables and sizes an indicate if have imaginary parts.
6. ; Semicolon: suppresses screen printing; also denotes a new row in an array.
7. : Colon: generates an array having regularly spaced elements.
8. , Comma: Separate elements of an array.
9. … Ellipsis : Continues a line.

Arrays (Vectors 1D)



To Create a row vector in Matlab, you simply type the elements inside a pair of square brackets, Separating the elements with a space or comma.

Variable_name=[value 1,value 2,value 3, … … .]

To Create a column vector, you can separate the elements by semicolons;

Variable_name=[value1;value 2;value 3; … …. .]

To Create a vector with constant spacing, by specifying the first, the last terms & spacing.

Variable_name=[m:q:n] where is m first term ,q step , n last term

Example :-

```
>> a= [1 , 7, 8, 6, -3]
>> b= [1; 6; 4; 9; 0]
>> c= [1:2:7]      c = 1 2 5 7
>> e= [1:2:4]
>>A= [4:2:1]
>> E= [4:-1:1]
```

- The linspace command creates a linearly spaced row vector, but instead you specify the number of values rather than the increment.

Variable_name = linspace(x1,x2,n)

Where x1, x2 are the first and last terms and n is the number of points.

e.g speed=linspace(5,8,31)

Note : If n is omitted, the number of points defaults to100.

Spacing =Difference between first and last terms/(n-1)

➢**The logspace command creates an array of logarithmically spaced elements.**

Variable_name = logspace(a,b,n)

Where n is the number of points between 10^a and 10^b

e.g >>displacment=logspace(-1,1,4)

displacment= [0.1000 0.4642 2.1544 10.0000]

Note: If n is omitted, the number of points defaults to 50.

Matrix (2 D)

An array having rows and columns is a two-dimensional array that is sometimes called a *matrix*.

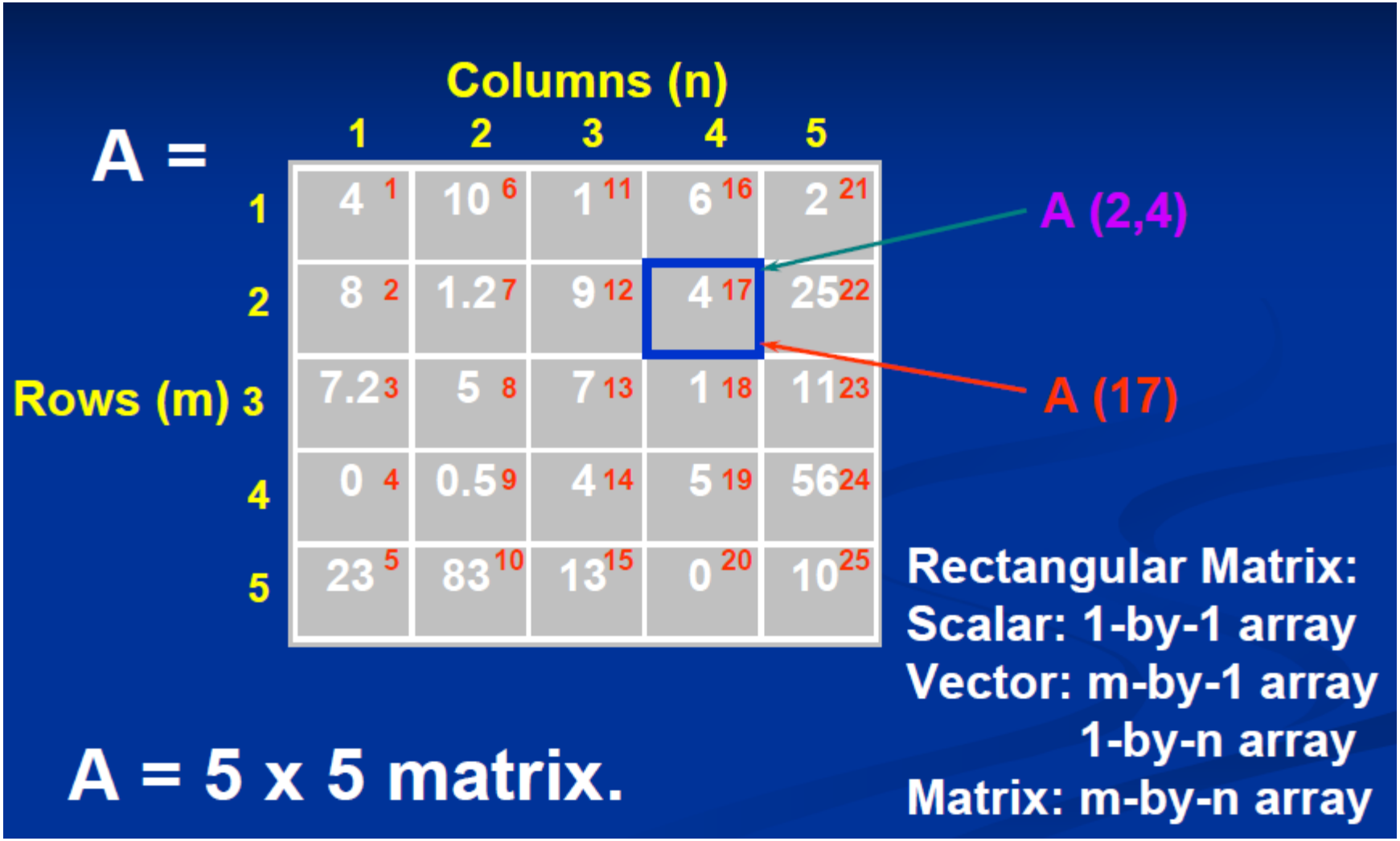
For Example:- The matrix M

$$M = \begin{bmatrix} 2 & 4 & 10 \\ 16 & 3 & 7 \\ 8 & 4 & 9 \\ 3 & 12 & 15 \end{bmatrix}$$

has four rows and three columns

***Vectors* are a special case of matrices having one row and one column**

The Matrix in Matlab



Creating Matrices

If the matrix is small you can type it row by row, separating the elements in a given row with space or commas (,) and separating the rows with semicolons (;). For example, typing

```
>>A = [2,4,10;16,3,7];
```

Creates the following matrix:

$$A = \begin{bmatrix} 2 & 4 & 10 \\ 16 & 3 & 7 \end{bmatrix}$$

Remember, Spaces or commas separate elements in different columns, whereas semicolons separate elements in different rows

Creating Matrices form Vectors

Suppose a = [1, 3, 5] and b = [7, 9, 11] (row vectors). Note the difference between the results given by [a b] or [a , b] and [a ; b] in the following:

```
>> a = [1, 3, 5] ;
>> b = [7, 9, 11] ;
>> c = [a b]  or c = [a , b]
c =
     1   3   5   7   9   11
>> D = [a ; b]
D =  1   3   5
       7   9   11
```

You not need to use symbols to create a new array.

***For example*, you can type**

```
>> D = [ [1, 3, 5] ; [7, 9, 11] ];  Note the double [ [ ] ]
```

Matrix Manipulation Functions

- zeros : Create an array/matrix of all zeros. e.g >> zeros(4,3)
- ones : Create an array/matrix of all ones. e.g >> ones(3,4)
- eye : Identity Matrix (square Matrix). e.g >> eye(3)
- rand : Uniformly distributed random numbers. e.g >> rand(2,3)
- diag : Diagonal matrices and diagonal of a matrix.
- length : Return the number of elements in vector.
- size : Return array dimensions.
- fliplr : Flip matrices left-right.
- flipud : Flip matrices up and down.
- (') : Transpose matrix.
- rot 90 : rotate matrix 90º.
- tril : Lower triangular part of a matrix.
- triu : Upper triangular part of a matrix.
- dot : Vector dot product.
- cross : Vector cross product.
- det : Matrix determinant.
- inv : Matrix inverse.
- eig: Evaluate eigenvalues and eigenvectors.

Array Addressing

The colon (:) operator selects individual elements, rows, columns, or ''subarrays" of arrays. *Here are some examples:*

- **v (:) represents all the row or column elements of the vector v.**
- **v (m:n) represents the elements m through n of the vector v.**
- **A (:) Represents all elements in matrix A.**
- **A (:, n) Refers to the elements in all the rows of column n of the matrix A.**
- **A (m, :) Refers to the elements in all the columns of row m of the matrix A.**
- **A (:,m:n) Refers to the elements in all the rows between columns m &n of the matrix A.**
- **A (m:n,:) Refers to the elements in all the columns between rows m &n of the matrix A.**
- **A (m:n,p:q) Refers to elements in rows m through n & columns p through q of the matrix A.**
- **A ([m,n],[p,q]) Refers to elements in rows m & n that intersect with elements in columns p & q.**
- **For deleting elements use the following A (~) = [ ]**

One can use array indices to extract a smaller array from another array, for example, if you first create the array B

$$B = \begin{bmatrix} 2 & 4 & 10 & 13 \\ 16 & 3 & 7 & 18 \\ 8 & 4 & 9 & 25 \\ 3 & 12 & 15 & 17 \end{bmatrix}$$

Then type C = B (2:3,1:3), you can produce the following array:

$$C = \begin{bmatrix} 16 & 3 & 7 \\ 8 & 4 & 9 \end{bmatrix}$$

Array Subscripting / Indexing