Experiment no: 02

Experiment name: Observation of Matrix operations in MATLAB

Objective: To perform matrix additions, subtractions and multiplications in MATLAB

Software Requirement: MATLAB 2014

Theory:

Matrices

The MATLAB environment uses the term matrix to indicate a variable containing real or complex numbers arranged in a two-dimensional grid. An array is, more generally, a vector, matrix, or higher dimensional grid of numbers. All arrays in MATLAB are rectangular, in the sense that the component vectors along any dimension are all the same length.

Adding and Subtracting Matrices

Addition and subtraction of matrices is defined just as it is for arrays, element by element. Adding A to B, and then subtracting A from the result recovers B:

```
A = pascal(3);

B = magic(3);

X = A + B

X =

     9     2     7

     4     7    10

     5    12     8

Y = X - A

Y =

     8     1     6

     3     5     7

     4     9     2
```

Vector Products and Transpose

A row vector and a column vector of the same length can be multiplied in either order. The result is either a scalar, the inner product, or a matrix, the outer product :

```
u = [3; 1; 4];
```

```
v = [2 0 -1];
x = v*u


x =
     2


X = u*v


X =
     6     0    -3
     2     0    -1
     8     0    -4
```

Transposition turns a row vector into a column vector:

```
x = v'


x =
     2
     0
    -1
```

If x and y are both real column vectors, the product x*y is not defined, but the two products

```
x'*y
```

and

```
y'*x
```

are the same scalar. This quantity is used so frequently, it has three different names: inner product, scalar product, or dot product.

Multiplying Matrices

Multiplication of matrices is defined in a way that reflects composition of the underlying linear transformations and allows compact representation of systems of simultaneous linear equations. The matrix product C = AB is defined when the column dimension of A is equal to the row dimension of B, or when one of them is a scalar.

MATLAB uses a single asterisk to denote matrix multiplication. The next two examples illustrate the fact that matrix multiplication is not commutative; AB is usually not equal to BA:

```
X = A*B

X =

    15    15    15

    26    38    26

    41    70    39

Y = B*A

Y =

    15    28    47

    15    34    60

    15    28    43
```

A matrix can be multiplied on the right by a column vector and on the left by a row vector:

```
u = [3; 1; 4];

x = A*u
```

```
x =

     8
    17
    30
```

```
v = [2 0 -1];
y = v*B
```

```
y =

    12    -7    10
```

Identity Matrix

Generally accepted mathematical notation uses the capital letter I to denote identity matrices, matrices of various sizes with ones on the main diagonal and zeros elsewhere. These matrices have the property that AI = A and IA = A whenever the dimensions are compatible. The original version of MATLAB could not use I for this purpose because it did not distinguish between uppercase and lowercase letters and i already served as a subscript and as the complex unit. So an English language pun was introduced. The function

```
eye(m,n)
```

returns an m-by-n rectangular identity matrix and eye(n) returns an n-by-n square identity matrix.

Lab Tasks:

1. Execute addition and subtraction of matrics

```
Command Window
>> A = [1 2 3;4 5 6;7 8 9]

A =

     1     2     3
     4     5     6
     7     8     9

>> B = [5 2 8;7 5 2;9 6 3]

B =

     5     2     8
     7     5     2
     9     6     3

>> X = A + B

X =

     6     4    11
    11    10     8
    16    14    12

>> Y = A- B

Y =

    -4     0    -5
    -3     0     4
    -2     2     6
```

2. Execute multiplication of matrics

```
Command Window
>> A = [1 2 3;4 5 6;7 8 9];
>> B = [2 5 8;3 6 9;7 5 3];
>> multi = A * B;
>> multi = A * B

multi =

    29    32    35
    65    80    95
   101   128   155

>>
```

3. Find out INVERSE MATRIX of Your Input A Matrix.

```
Command Window
>> A = [2 4 5; 1 2 3; 4 5 9]

A =

     2     4     5
     1     2     3
     4     5     9

>> invs = inv(A)

invs =

    1.0000   -3.6667    0.6667
    1.0000   -0.6667   -0.3333
   -1.0000    2.0000         0

>>
```

Discussion:

In this lab, we focused on matrix implementation and key operations—addition, subtraction, multiplication, and inversion. The hands-on exercises, executed error-free, demonstrated proficiency in utilizing MATLAB for efficient matrix manipulations. This foundational understanding lays the groundwork for applying matrices in various mathematical and engineering applications.