Essentials of Analytical Geometry and Linear Algebra I, Class #3

Innopolis University, September 2023

Operation with matrices

1. Let $A = \begin{bmatrix} 3 & 1 \\ 5 & -2 \end{bmatrix}$, $B = \begin{bmatrix} -2 & 1 \\ 3 & 4 \end{bmatrix}$, $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$:
   a) Find $A + B$;
   b) Find $2A - 3B + I$;
   c) Find AB and BA (make sure that, in general, $AB \neq BA$ for matrices);
   d) Find AI and IA.

```
import sympy as sp
A = sp.Matrix([[3, 1],[5, -2]])
B = sp.Matrix([[-2,1],[3,4]])

# a)
res1 = A + B
print("a) ")
res1
```

a)

$$\begin{bmatrix} 1 & 2 \\ 8 & 2 \end{bmatrix}$$

```
print("b)")
res2 = 2*A - 3*B + sp.eye(2)
res2
```

b)

$$\begin{bmatrix} 13 & -1 \\ 1 & -15 \end{bmatrix}$$

```
# c)
res3a = A*B
print("c_a)")
res3a
```

c_a)

$$\begin{bmatrix} -3 & 7 \\ -16 & -3 \end{bmatrix}$$

```
res3b = B*A
print("c_b)")
res3b
```

c_b)

$$\begin{bmatrix} -1 & -4 \\ 29 & -5 \end{bmatrix}$$

```
print("d_a)")
res4a = A*sp.eye(2)
res4a
```

d_a)

$$\begin{bmatrix} 3 & 1 \\ 5 & -2 \end{bmatrix}$$

```
print("d_b)")
res4b = sp.eye(2) * A
res4b
```

d_b)

$$\begin{bmatrix} 3 & 1 \\ 5 & -2 \end{bmatrix}$$

2. Let $A = \begin{bmatrix} 2 & -1 & -1 \end{bmatrix}$ and $B = \begin{bmatrix} -2 \\ -1 \\ 3 \end{bmatrix}$:

a) Find AB и BA, if exists;
b) Find $A^T B$ и BA^T, if exists.

```
import sympy as sp
A = sp.Matrix([[2],[-1],[-1]]).T
B = sp.Matrix([[-2], [-1], [3]])

print("a_a)")
t1a = A*B
t1a
```

a_a)
$$\begin{bmatrix} -6 \end{bmatrix}$$

```
t1b = B*A
print("a_b)")
t1b
```

a_b)
$$\begin{bmatrix} -4 & 2 & 2 \\ -2 & 1 & 1 \\ 6 & -3 & -3 \end{bmatrix}$$

t2a = A.T*B (cannot be solved)

t2b = B*A.T (cannot be solved)

3. If solution exists, what the dimension of the result matrix. There are several matrices: $\underset{3\times3}{A}, \underset{2\times3}{B}, \underset{3\times2}{C}, \underset{3\times5}{D}, \underset{1\times2}{E}, \underset{3\times1}{K}$.

a) ABC;
b) AB^TC^T;
c) $EBAE$;
d) $K^T \times K^TCE^T$.

```
import sympy as sp
A = sp.MatrixSymbol('A',3,3)
B = sp.MatrixSymbol('B',2,3)
C = sp.MatrixSymbol('C',3,2)
D = sp.MatrixSymbol('D',3,5)
E = sp.MatrixSymbol('E',1,2)
K = sp.MatrixSymbol('K',3,1)

print("a)")
res1 = A*B*C
res1
```

```
a)
---------------------------------------------------------------------------
--
ShapeError                                Traceback (most recent call
last)
<ipython-input-119-5e9b210d33fb> in <cell line: 10>()
      8
      9 print("a)")
---> 10 res1 = A*B*C
     11 res1
```

4 frames

```
/usr/local/lib/python3.10/dist-
packages/sympy/matrices/expressions/_shape.py in
validate_matmul_integer(*args)
    100         i, j = A.cols, B.rows
    101         if isinstance(i, (int, Integer)) and isinstance(j, (int,
Integer)) and i != j:
--> 102             raise ShapeError("Matrices are not aligned", i, j)
```

```
print("b)")
res2 = A*B.T*C.T
```

```
res2.shape
```

```
b)
(3, 3)
```

```
print("c)")
res3 = E*B*A*E
res3.shape
```

```
c)
---------------------------------------------------------------------------
ShapeError                                Traceback (most recent call last)
<ipython-input-121-1a6ed466ca13> in <cell line: 2>()
      1 print("c)")
----> 2 res3 = E*B*A*E
      3 res3.shape

 4 frames
/usr/local/lib/python3.10/dist-packages/sympy/matrices/expressions/_shape.py in validate_matmul_integer(*args)
    100         i, j = A.cols, B.rows
    101         if isinstance(i, (int, Integer)) and isinstance(j, (int, Integer)) and i != j:
--> 102             raise ShapeError("Matrices are not aligned", i, j)
```

```
print("d)")

res4 = sp.Matrix.cross(sp.Matrix(K.T),sp.Matrix(K.T))*C*E.T
res4.shape
```

```
d)
(1, 1)
```

Determinants

1. Find the determinants of the following matrices:

a) $A = \begin{bmatrix} 5 & -2 \\ 1 & 6 \end{bmatrix}$;

b) $B = \begin{bmatrix} 1 & -3 & -1 \\ -2 & 7 & 2 \\ 3 & 2 & -4 \end{bmatrix}$.

```
import sympy as sp
A = sp.Matrix([[5, -2],[1, 6]])
B = sp.Matrix([[1,-3, -1],[-2,7,2], [3,2,-4]])

print("a)", A.det())
print("b)", B.det())
```

```
a) 32
b) -1
```

2. A triangle is constructed on vectors \textbf{a}= $\begin{bmatrix} 2 \\ 4 \\ -1 \end{bmatrix}$ and \textbf{b}= $\begin{bmatrix} -2 \\ 1 \\ 1 \end{bmatrix}$

   a) Find the area of this triangle.

   b) Find the altitudes of this triangle.

```
import sympy as sp
a = sp.Matrix([[2],[4],[-1]])
b = sp.Matrix([[-2], [1], [1]])
area = sp.Matrix.cross(a,b).norm()/2
print("a)")
area
```

a)

$$\frac{5\sqrt{5}}{2}$$

$S = \frac{1}{2}ah$

```
print("b_a)")
h= 2*area / a.norm()
h
```

b_a)
$\frac{5\sqrt{105}}{21}$

3. Find the matrix product AB, if $A = \begin{bmatrix} 1 & 2 & 5 \\ 3 & 7 & x \end{bmatrix}$, $B = \begin{bmatrix} 5 & -1 \\ x & 2 \\ -3 & -1 \end{bmatrix}$.

Then find the largest possible value of $det(AB)$.

```
import sympy as sp
x = sp.Symbol('x')
A = sp.Matrix([[1,2,5],[3,7,x]])
B = sp.Matrix([[5,-1],[x,2],[-3,-1]])

print("a)")
resa = A*B
resa
```

a)
$\begin{bmatrix} 2x - 10 & -2 \\ 4x + 15 & 11 - x \end{bmatrix}$

```
print("b)")
det_resa = resa.det()
det_resa
```

b)
$-2x^2 + 40x - 80$

```
ddet_resa = sp.Derivative(det_resa)
ddet_resa = ddet_resa.doit()
ddet_resa
```

$40 - 4x$

```
resb = sp.solve(ddet_resa,x)
resb
```

```
[10]
```

Scalar Triple Product

1. Find the scalar triple product of $\mathbf{a} = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 7 \\ 3 \\ -5 \end{bmatrix}$, $\mathbf{c} = \begin{bmatrix} 3 \\ 4 \\ -3 \end{bmatrix}$.

```
import sympy as sp
a = sp.Matrix([[1],[2],[-1]])
b = sp.Matrix([[7],[3],[-5]])
c = sp.Matrix([[3],[4],[-3]])

res = a.dot(b.cross(c))
res
```

4

2. Vectors $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$ are not coplanar. Find all values of θ such that vectors $\mathbf{a} + 2\mathbf{b} + \theta\mathbf{c}$, $4\mathbf{a} + 5\mathbf{b} + 6\mathbf{c}$, $7\mathbf{a} + 8\mathbf{b} + \theta^2\mathbf{c}$ are coplanar.

```
import sympy as sp
a = sp.MatrixSymbol("a",3,1)
b = sp.MatrixSymbol("b",3,1)
c = sp.MatrixSymbol("c",3,1)
```

```python
theta = sp.Symbol("theta")

# The three vectors are coplanar if their scalar triple product is zero.

i = a + 2*b + theta*c
j = 4*a + 5*b + 6*c
k = 7*a + 8*b + theta**2 * c

matri = sp.Matrix([sp.Matrix(i).T,sp.Matrix(j).T,sp.Matrix(k).T]).T
resres = sp.solve(matri.det(),theta)
resres
```

```
[-4, 3]
```