5.13.27

EE25BTECH11001 - Aarush Dilawri

October 11, 2025

Question: Let $\mathbf{A} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ and $\mathbf{B} = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}$, $a, b \in \mathbb{N}$.

(a) there cannot exist any $\mathbf{B}$ such that $\mathbf{AB} = \mathbf{BA}$

(b) there exist more than one but finite number of $\mathbf{B}$ such that $\mathbf{AB} = \mathbf{BA}$

(c) there exists exactly one $\mathbf{B}$ such that $\mathbf{AB} = \mathbf{BA}$

(d) there exist infinitely many $\mathbf{B}$ such that $\mathbf{AB} = \mathbf{BA}$

Solution

Solution:

$$\text{Let } \mathbf{A} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}, \quad a, b \in \mathbb{N}. \tag{1}$$

We compute **AB**:

$$\mathbf{AB} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \tag{2}$$

$$= \begin{pmatrix} a & 2b \\ 3a & 4b \end{pmatrix}. \tag{3}$$

Solution

Similarly, compute **BA**:

$$\mathbf{BA} = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \tag{4}$$

$$= \begin{pmatrix} a & 2a \\ 3b & 4b \end{pmatrix}. \tag{5}$$

Solution

For $\mathbf{AB} = \mathbf{BA}$, we must have:

$$\begin{pmatrix} a & 2b \\ 3a & 4b \end{pmatrix} = \begin{pmatrix} a & 2a \\ 3b & 4b \end{pmatrix}. \tag{6}$$

Equating the corresponding entries gives:

$$2b = 2a \implies b = a, \tag{7}$$

$$3a = 3b \implies a = b. \tag{8}$$

Solution

Hence,

$$\mathbf{B} = \begin{pmatrix} a & 0 \\ 0 & a \end{pmatrix} = a\mathbf{I}. \tag{9}$$

Since $a \in \mathbb{N}$, there are infinitely many such $\mathbf{B}$.
Therefore, the answer is (d) there exist infinitely many $\mathbf{B}$ such that $\mathbf{AB} = \mathbf{BA}$.

C Code (code.c)

```c
#int countCommutingMatrices(int n, int *A, int maxVal) {
    // if any off-diagonal entry of A is non-zero, infinite solutions (a=b
        )
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            if (i != j && A[i*n + j] != 0)
                return -1;

    // otherwise finite: count diagonal Bs with a=b
    return maxVal;
}
```

Python Code (code.py)

```
def count_commuting(A, n, maxVal):
    for i in range(n):
        for j in range(n):
            if i != j and A[i][j] != 0:
                return -1
    return maxVal

A = [[1, 2],
     [3, 4]]
n = 2
maxVal = 5

res = count_commuting(A, n, maxVal)
print("Number of commuting matrices B = (infinite)" if res == -1 else f
    "Number of B = {res}")
```

Python Code (nativecode.py)

```python
import ctypes

lib = ctypes.CDLL("./code.so")
lib.countCommutingMatrices.argtypes = [ctypes.c_int, ctypes.POINTER(
    ctypes.c_int), ctypes.c_int]
lib.countCommutingMatrices.restype = ctypes.c_int

n = 2
A = [1, 2, 3, 4]
maxVal = 5
A_c = (ctypes.c_int * (n*n))(*A)

res = lib.countCommutingMatrices(n, A_c, maxVal)
print("Number of commuting matrices B = (infinite)" if res == -1 else f
    "Number of B = {res}")
```