12.686

EE25BTECH11049-Sai Krishna Bakki

October 12, 2025

Question

$\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are vectors of length 4.

$$\mathbf{A} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix}, \mathbf{B} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix}, \mathbf{C} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix}, \mathbf{D} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{pmatrix}$$

It is known that $\mathbf{B}$ is not a scalar multiple of $\mathbf{A}$. Also, $\mathbf{C}$ is linearly independent of $\mathbf{A}$ and $\mathbf{B}$. Further, $\mathbf{D} = 3\mathbf{A} + 2\mathbf{B} + \mathbf{C}$. The rank of the matrix $\begin{pmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ a_4 & b_4 & c_4 & d_4 \end{pmatrix}$ is

Theoretical Solution

The rank of the matrix is defined as the number of linearly independent columns or rows it contains. Let's analyze the linear independence of the columns of the given matrix, which are the vectors $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$.

Given:

1. $\mathbf{A}, \mathbf{B}$ are linearly independent
2. $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are linearly independent
3. $\mathbf{D} = 3\mathbf{A} + 2\mathbf{B} + \mathbf{C}$ where $\mathbf{D}$ is linearly dependent on $\mathbf{A}, \mathbf{B}, \mathbf{C}$

$$\mathbf{D} = \begin{pmatrix} 3a_1 + 2b_1 + c_1 \\ 3a_2 + 2b_2 + c_2 \\ 3a_3 + 2b_3 + c_3 \\ 3a_4 + 2b_4 + c_4 \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{pmatrix} \tag{1}$$

$$\mathbf{M} = \begin{pmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ a_4 & b_4 & c_4 & d_4 \end{pmatrix} \tag{2}$$

(3)

Theoretical Solution

$$\mathbf{M} = \begin{pmatrix} a_1 & b_1 & c_1 & 3a_1 + 2b_1 + c_1 \\ a_2 & b_2 & c_2 & 3a_2 + 2b_2 + c_2 \\ a_3 & b_3 & c_3 & 3a_3 + 2b_3 + c_3 \\ a_4 & b_4 & c_4 & 3a_4 + 2b_4 + c_4 \end{pmatrix} \tag{4}$$

There is one linearly dependent column so there are three linearly independent columns.

$\therefore$ The rank of $\mathbf{M}$ is 3.

C Code

```c
    #include <stdio.h>

// Define the length of the vectors.
#define VECTOR_LENGTH 4

void calculate_D(const int* A, const int* B, const int* C, int* D
    ) {
    for (int i = 0; i < VECTOR_LENGTH; i++) {
        D[i] = (3 * A[i]) + (2 * B[i]) + C[i];
    }
}
```

Python Code Through Shared Output

```
import ctypes
import random
import os
# Determine the library file name based on the OS
lib_name = 'matrix_ops.so' if os.name != 'nt' else 'matrix_ops.
    dll'
lib_path = os.path.join(os.path.dirname(os.path.abspath(__file__)
    ), lib_name)

# 1. Load the shared library
try:
    c_lib = ctypes.CDLL(lib_path)
except OSError as e:
    print(fError loading shared library: {e})
    print(Please make sure you have compiled `matrix_ops.c` into
        a shared library.)
    exit()
```

Python Code Through Shared Output

```
# 2. Define the function signature (argument types and return
    type)
# The function is: void calculate_D(int* A, int* B, int* C, int*
    D)
# We need to specify that the arguments are pointers to integers.
IntArray4 = ctypes.c_int * 4
c_lib.calculate_D.argtypes = [
    ctypes.POINTER(ctypes.c_int),
    ctypes.POINTER(ctypes.c_int),
    ctypes.POINTER(ctypes.c_int),
    ctypes.POINTER(ctypes.c_int)
]
c_lib.calculate_D.restype = None # The C function returns void

# 3. Prepare the data in Python
# Generate random vectors A, B, and C
A_py = [random.randint(1, 10) for _ in range(4)]
B_py = [random.randint(1, 10) for _ in range(4)]
C_py = [random.randint(1, 10) for _ in range(4)]
```

Python Code Through Shared Output

```
# Convert Python lists to C-compatible integer arrays
A_c = IntArray4(*A_py)
B_c = IntArray4(*B_py)
C_c = IntArray4(*C_py)

# Create an empty C array to store the result D
D_c = IntArray4()

# 4. Call the C function
c_lib.calculate_D(A_c, B_c, C_c, D_c)

# 5. Convert the result back to a Python list to display it
D_py = list(D_c)

# Print the results
print(--- Using C function from Python (ctypes) ---)
print(fVector A: {A_py})
print(fVector B: {B_py})
print(fVector C: {C_py})
```

Python Code Through Shared Output

```
print(fResult D (3*A + 2*B + C): {D_py})

# Verification
D_verify = [(3 * a + 2 * b + c) for a, b, c in zip(A_py, B_py,
    C_py)]
print(fVerification in Python: {D_verify})
assert D_py == D_verify
print(Result matches Python's calculation.)
```

Python Code

```
import numpy as np
import numpy.linalg as LA

# 1. Define three linearly independent vectors A, B, and C of
    length 4.
# A simple choice is to use vectors with a single non-zero
    element.
A = np.array([[1], [0], [0], [0]])
B = np.array([[0], [1], [0], [0]])
C = np.array([[0], [0], [1], [0]])

# 2. Define vector D as a linear combination of A, B, and C, as
    per the problem.
# D = 3A + 2B + C
D = 3 * A + 2 * B + C

# 3. Construct the matrix M by combining the vectors as columns.
M = np.concatenate((A, B, C, D), axis=1)
```

Python Code

```
# 4. Calculate the rank of the matrix M.
# The rank is the maximum number of linearly independent columns.
# Since D is dependent on A, B, and C, the rank should be 3.
rank_of_M = LA.matrix_rank(M)

# Print the vectors, the resulting matrix, and its rank for
    verification.
print(--- VECTORS ---)
print(Vector A:\n, A)
print(\nVector B:\n, B)
print(\nVector C:\n, C)
print(\nVector D = 3A + 2B + C:\n, D)
print(\n--- MATRIX [A B C D] ---)
print(M)
print(\n--- RESULT ---)
print(fThe rank of the matrix is: {rank_of_M})
```