MATGEO 1-1.4-12

Sai Akhila
Dept. of Electrical Engg.,
IIT Hyderabad.

November 6, 2024

1. Problem

2. Solution
   - Given values
   - Section formula
   - Verified values

Problem Statement

The position vector of the point which divides the join of points $2a - 3b$ and $a + b$ in the ratio $3 : 1$ is:

Given values

Vector	Coordinates
A	$2a - 3b$
B	$a + b$
C	$\frac{5}{4}\ a$

Table: Given Values

Section formula

Solution: Using Section Formula ($k = 3$):

$$C = \frac{kB + A}{k + 1} \tag{3.1}$$

$$C = \frac{1}{3 + 1}(3B + A) \tag{3.2}$$

$$\implies C = \frac{1}{4}((3a + 3b) + (2a - 3b)) \tag{3.3}$$

$$C = \frac{5}{4}a \tag{3.4}$$

The code in

`https://github.com/SaiAkhila326/Mt/blob/master/mt/q1m/codes/verify.py`

verifies the equation.

Verified values

Row	a	b	A = 2a - 3b	B = a + b	Resultant Vector
Row 1	1.00	3.00	-7.00	4.00	1.25
Row 2	2.00	4.00	-8.00	6.00	2.50

Table: Verified values

C Code I

```c
#include <stdio.h>
#include <stdlib.h>


double **createMat(int m,int n);
double **Matadd(double **a, double **b, int m, int n);//add two matrices
double **Matscale(double **a, int m, int n, double k);
double **Matsec(double **a, double ** b, int m, double k);

/*int main(void)
{

    int m = 2;
    double k = 3.0;


    double **a = (double **)malloc(m * sizeof(double *));
    double **b = (double **)malloc(m * sizeof(double *));
    double **c = (double **)malloc(m*sizeof(double *));
```

C Code II

```
20      double **d = (double **)malloc(m*sizeof(double *));
21      for (int i = 0; i < m; i++) {
22          a[i] = (double *)malloc(1 * sizeof(double));
23          b[i] = (double *)malloc(1 * sizeof(double));
24          c[i]=(double *)malloc(1*sizeof(double));
25          d[i]=(double *)malloc(1*sizeof(double));
26      }
27
28
29
30      for (int i = 0; i < m; i++) {
31          a[i][0] = i + 1;
32          b[i][0] = (i + 1) * 2;
33      }
34
35      c= Matadd(Matscale(a,m,1,2),Matscale(b,m,1,-3),m,1);
36      d= Matadd(Matscale(a,m,1,1),Matscale(b,m,1,1),m,1);
37      //for(int i=0;i<m;i++)
38
39     //   printf("%lf %lf",c[i][0],d[i][0]);
```

C Code III

```
40
41
42      double **temp = Matsec(c, d, m, k);
43
44      FILE *file = fopen("output.txt", "w");
45      if (file == NULL) {
46          printf("Error opening the file!\n");
47          return 1;
48      }
49
50
51
52      for (int i = 0; i < m; i++) {
53  fprintf(file, "%lf ", *a[i]);
54          fprintf(file, "%lf \n", temp[i][0]);
55      }
56
57
58      fclose(file);
59
```

C Code IV

```
60      for (int i = 0; i < m; i++) {
61          free(a[i]);
62          free(b[i]);
63          free(temp[i]);
64      }
65      free(a);
66      free(b);
67      free(temp);
68
69      return 0;
70  }*/
71
72
73
74
75
76
77  double **createMat(int m,int n)
78  {
79   int i;
```

C Code V

```
80  double **a;
81
82
83 a = (double **)malloc(m * sizeof( *a));
84     for (i=0; i<m; i++)
85          a[i] = (double *)malloc(n * sizeof( *a[i]));
86
87  return a;
88 }
89
90 double **Matsec(double **a, double ** b, int m, double k){
91   double **temp=createMat(m,1);
92   temp = Matscale(Matadd(a,Matscale(b,m,1,k),m,1),m,1,1/(k+1));
93   return temp;
94 }
95 double **Matadd(double **a,double **b, int m, int n){
96 int i, j;
97 double **c;
98 c = createMat(m,n);
99
```

C Code VI

```
00  for(i=0;i<m;i++)
01  {
02   for(j=0;j<n;j++)
03   {
04 c[i][j]= a[i][j]+b[i][j];
05   }
06  }
07 return c;
08 }
09 double **Matscale(double **a, int m, int n, double k){
10 int i, j;
11 double **c;
12 c = createMat(m,n);
13
14  for(i=0;i<m;i++)
15  {
16   for(j=0;j<n;j++)
17   {
18 c[i][j]= k*a[i][j];
19   }
```

C Code VII

```
20  }
21 return c;
22 }
```

Python Code I

```
1  import ctypes
2  import numpy as np
3
4  # Step 1: Load the shared object (op.so) file
5  example = ctypes.CDLL('/home/sai-akhila/Desktop/q1m/codes/op.so')
6
7  # Step 2: Define the argument types and return types for the
       required C functions
8  example.Matscale.argtypes = [
9      ctypes.POINTER(ctypes.POINTER(ctypes.c_double)),
       ctypes.c_int, ctypes.c_int, ctypes.c_double
10 ]
11 example.Matscale.restype =
       ctypes.POINTER(ctypes.POINTER(ctypes.c_double))
12
13 example.Matadd.argtypes = [
```

Python Code II

```
14      ctypes.POINTER(ctypes.POINTER(ctypes.c_double)),
        ctypes.POINTER(ctypes.POINTER(ctypes.c_double)),
        ctypes.c_int, ctypes.c_int
15  ]
16  example.Matadd.restype =
        ctypes.POINTER(ctypes.POINTER(ctypes.c_double))
17
18  example.Matsec.argtypes = [
19      ctypes.POINTER(ctypes.POINTER(ctypes.c_double)),
        ctypes.POINTER(ctypes.POINTER(ctypes.c_double)),
        ctypes.c_int, ctypes.c_double
20  ]
21  example.Matsec.restype =
        ctypes.POINTER(ctypes.POINTER(ctypes.c_double))
22
23  # Step 3: Helper function to convert a numpy array into a ctypes
        2D array
24  def create_2d_array(arr):
```

Python Code III

```
25      arr_ctypes = (ctypes.POINTER(ctypes.c_double) *
        arr.shape[0])()
26      for i in range(arr.shape[0]):
27          arr_ctypes[i] =
        arr[i].ctypes.data_as(ctypes.POINTER(ctypes.c_double))
28      return arr_ctypes
29
30  # Step 4: Define input arrays 'a' and 'b' (2x1 matrices)
31  a = np.array([[1], [2]], dtype=np.float64)  # Vector a: [[1],
        [2]]
32  b = np.array([[3], [4]], dtype=np.float64)  # Vector b: [[3],
        [4]]
33  m = 2                                       # Number of rows in
        a and b
34  n = 1                                       # Columns in vectors
        (1-column vectors)
35  k = 3.0                                     # Scalar for Matsec
        function
```

Python Code IV

```
36
37 # Convert the numpy arrays into ctypes arrays
38 a_ctypes = create_2d_array(a)
39 b_ctypes = create_2d_array(b)
40
41 # Step 5: Create vectors A = 2a - 3b and B = a + b using
       Matscale and Matadd
42
43 # A = 2a - 3b
44 scaled_a_2 = example.Matscale(a_ctypes, m, n, 2.0)  # 2a
45 scaled_b_3 = example.Matscale(b_ctypes, m, n, -3.0) # -3b
46 A_ctypes = example.Matadd(scaled_a_2, scaled_b_3, m, n)  # A =
       2a - 3b
47
48 # B = a + b
49 B_ctypes = example.Matadd(a_ctypes, b_ctypes, m, n)  # B = a + b
50
51 # Step 6: Call the C function Matsec on A and B
```

Python Code V

```
result_ctypes = example.Matsec(A_ctypes, B_ctypes, m, k)

# Step 7: Convert the result back to a numpy array for easier
    handling
result_np = np.zeros((m, 1), dtype=np.float64)
for i in range(m):
    result_np[i][0] = result_ctypes[i][0]

# Step 8: Generate LaTeX Table

def array_to_latex_row(arr):
    return " & ".join(map(lambda x: f"{x:.2f}", arr)) + " \\\\"

latex_table = r"""
\documentclass{article}
\usepackage{amsmath}
\usepackage{booktabs}
\usepackage{array}
```

Python Code VI

```
\begin{document}

\begin{table}[h!]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
\textbf{Row} & \textbf{a} & \textbf{b} & \textbf{A = 2a - 3b} &
    \textbf{B = a + b} & \textbf{Resultant Vector} \\
\hline
"""

for i in range(m):
    latex_table += f"Row {i+1} & {a[i][0]:.2f} & {b[i][0]:.2f} &
    {A_ctypes[i][0]:.2f} & {B_ctypes[i][0]:.2f} &
    {result_np[i][0]:.2f} \\\\\n"

latex_table += r"""
\hline
```

Python Code VII

```
\end{tabular}
\caption{Vectors a, b, A, B, and Result from Matsec function}
\end{table}

\end{document}
"""

# Step 9: Write LaTeX table to a .tex file
with open('output_table.tex', 'w') as f:
    f.write(latex_table)

print("LaTeX table written to output_table.tex")
```