Mobile Robotics

Lab/task: Lab 2

Daniel Rios

Date: 02/06/20

Introduction

The purpose of this lab is to familiarize ourselves with MATLAB as well as review linear algebra. MATLAB is a numerical computing environment, which is useful for solving systems of equations and graphing linear systems. This assignment served as a review and a tutorial on how to use MATLAB. We learned the basic commands on how to input multiple linear equations as matrices and learned how to perform operations on them. We also had the opportunity to show these matrices as vectors and were able to graph them to show a visual representation of the solution to the system. We also found that linear algebra can be used to encrypt messages.

Summary of Results

We were able to solve these linear systems and manipulate their matrix forms. Below are the solutions to the problems and the MATLAB code necessary to solve the problem. We were also tasked with using Linear Algebra to encrypt and decrypt messages. We were successful at this and below are the results.

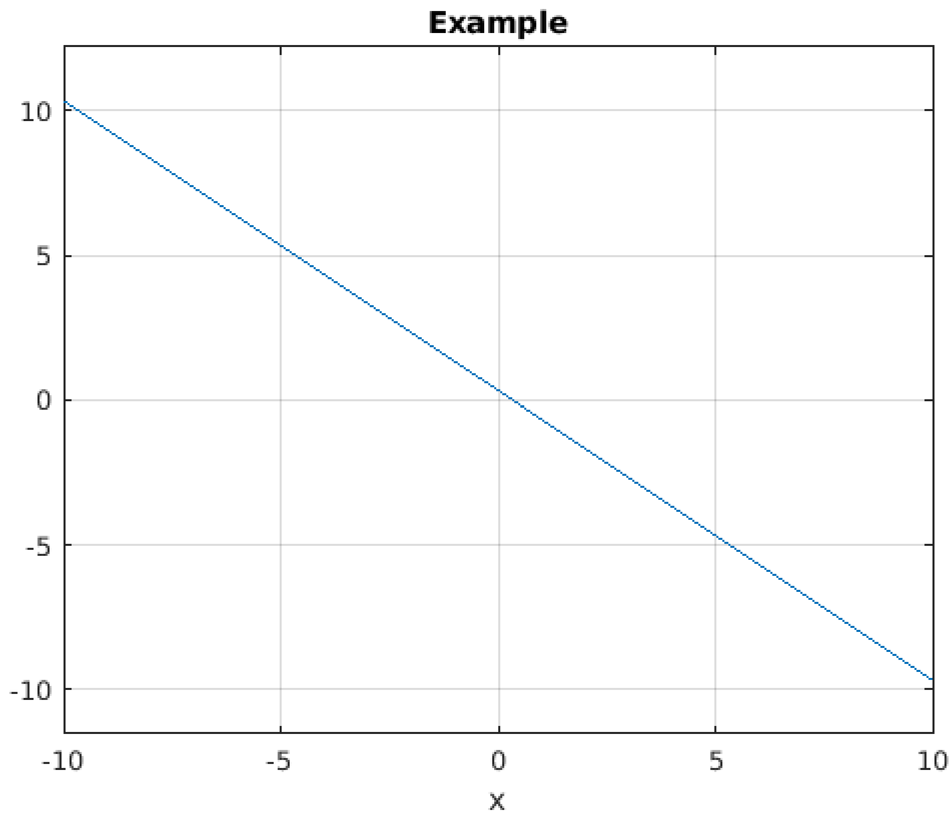
Conclusion

This lab assignment showed us how we can use computer aids to solve mathematical problems as well as allowed us to review the math necessary for this course.

Problem 1.)

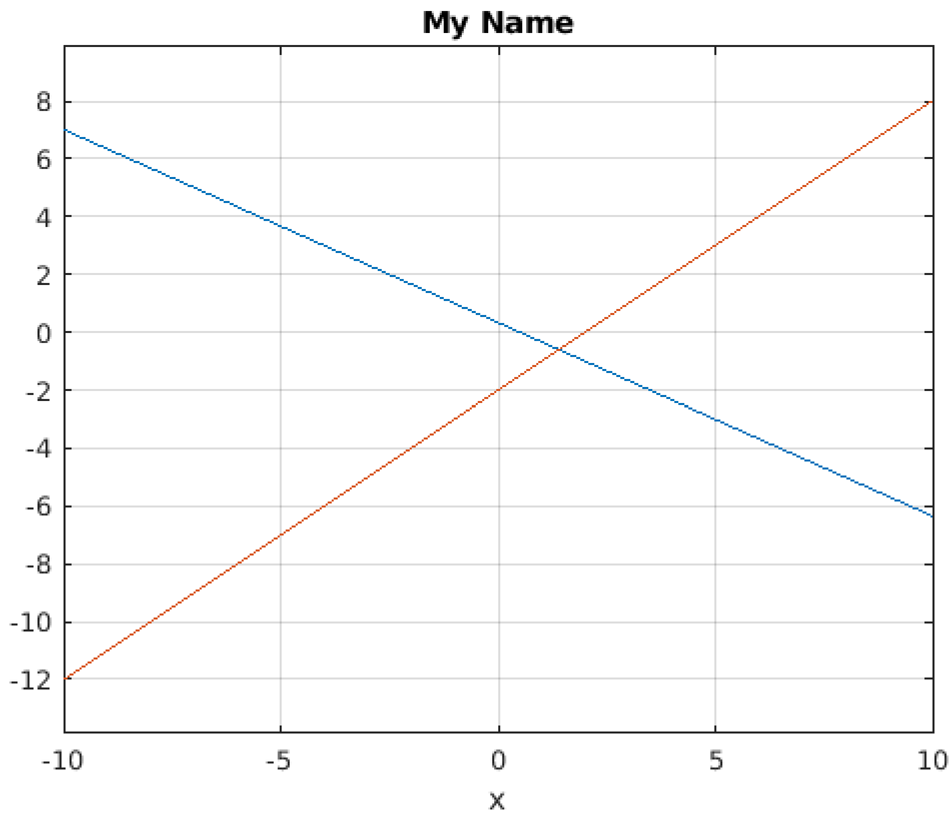
Example 1

```
ezplot('1/3 - 2*x/2',[-10,10])
title ('Example')
grid
```



Example 2

```
ezplot('1/3 - 2*x/3', [-10,10])
hold on
ezplot('x-2', [-10,10])
title('My Name')
grid
hold off
```



Number 1

```
A = [2 3; 1 -1]
```

```
A = 2×2
     2     3
     1    -1
```

```
b = [1;2]
```

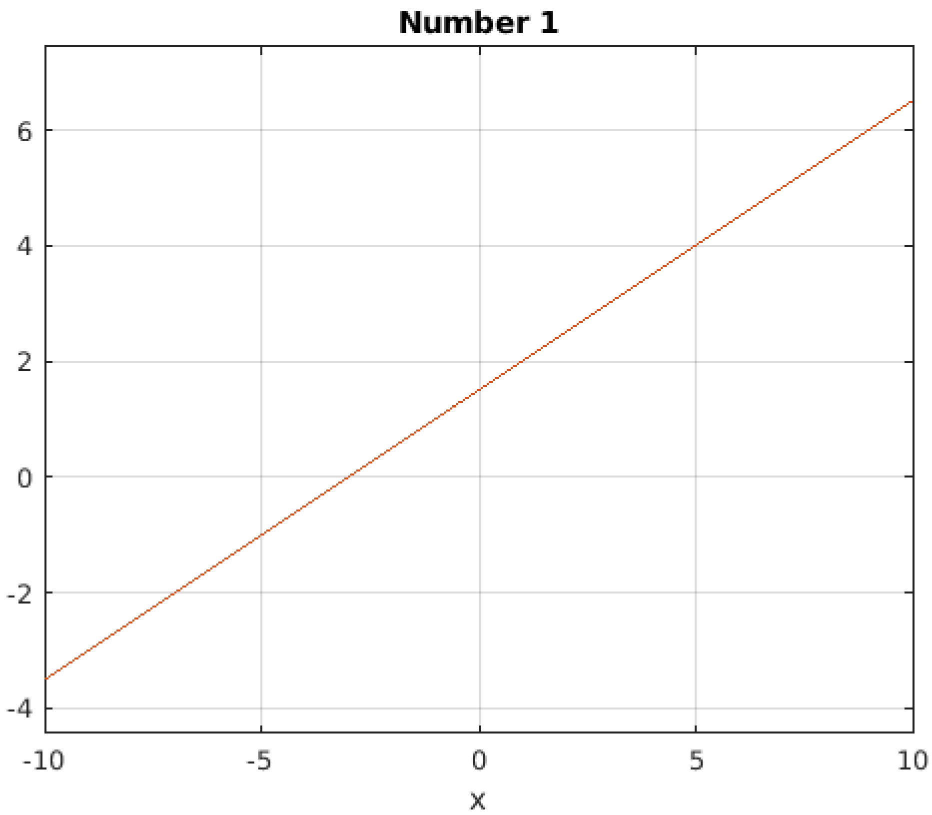
```
b = 2×1
     1
     2
```

```
A\b
```

```
ans = 2×1
    1.4000
   -0.6000
```

1.) Graph, They are on the same line.

```
ezplot('x/2 + 3/2',[-10,10])
hold on
ezplot('.5*x + 3/2',[-10,10])
title('Number 1')
grid
hold off
```



Infinitely Many Solutions. The Lines are overlapped and linearly dependent.

```
A = [-1 2 ; 2 -4]
```

```
A = 2×2
    -1     2
     2    -4
```

```
b = [3 ; -6]
```

```
b = 2×1
     3
    -6
```

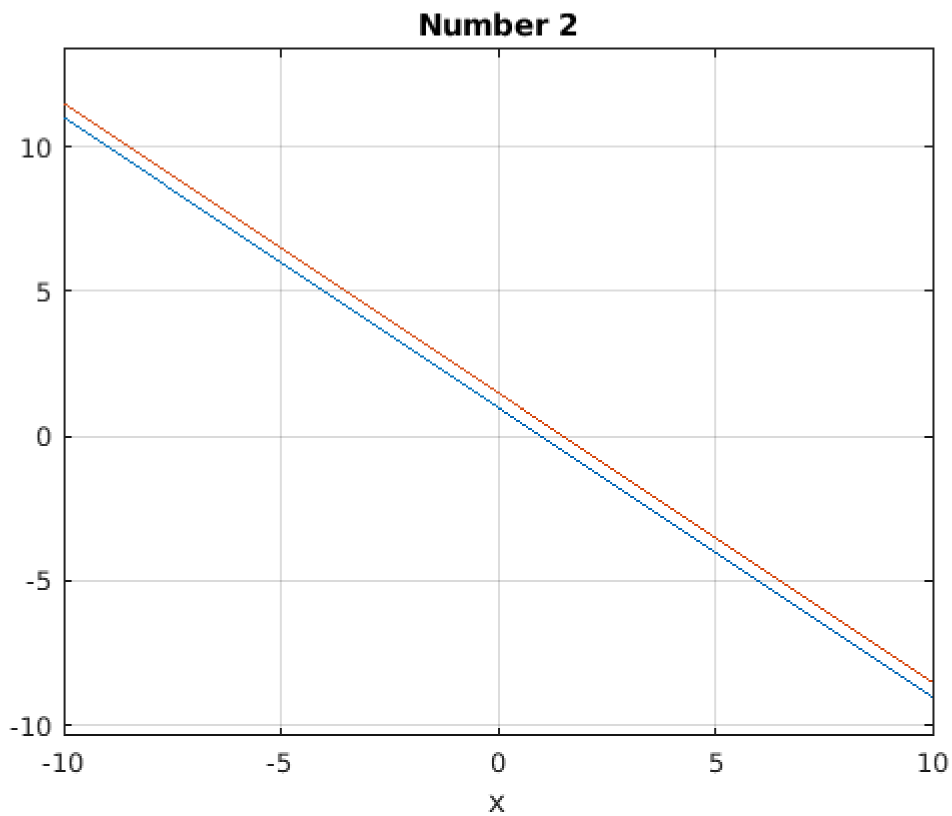
```
A\b
```

```
Warning: Matrix is singular to working precision.
ans = 2×1
   NaN
   NaN
```

2.) Graph

```
ezplot('1 - x',[-10,10])
hold on
ezplot('3/2 - x',[-10,10])
title ('Number 2')
grid
```

```
hold off
```



No Solution, They do not intersect.

```
A = [1 1 ; 2 2]
```

```
A = 2×2
     1     1
     2     2
```

```
b = [1 ; 3]
```

```
b = 2×1
     1
     3
```

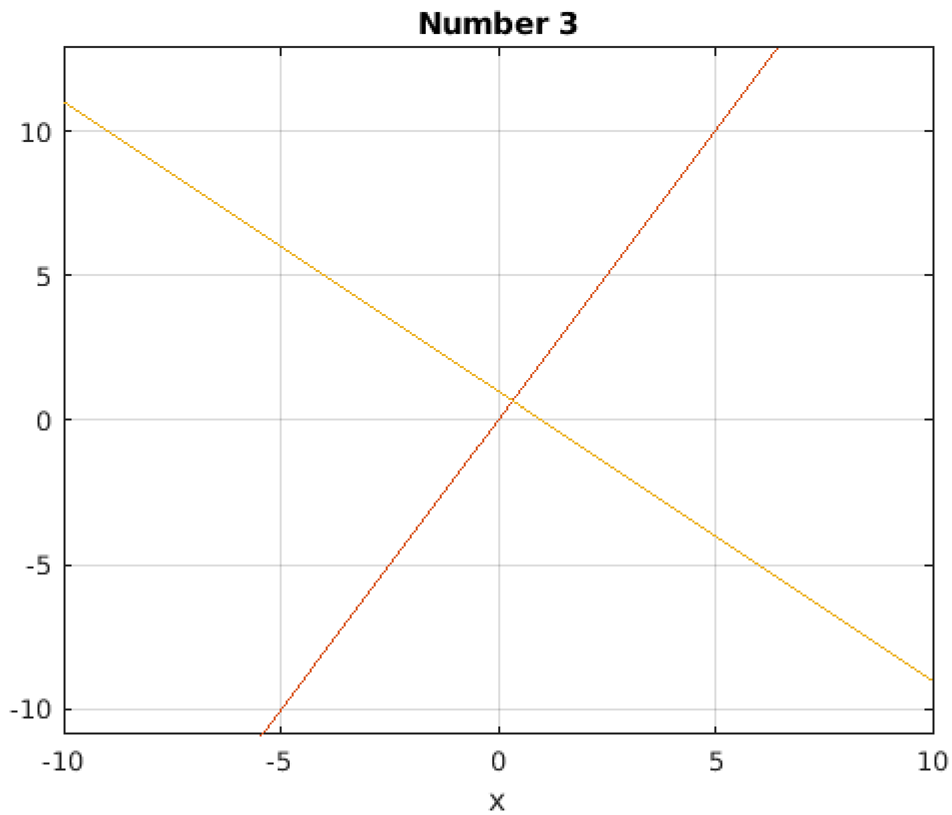
```
A\b
```

```
Warning: Matrix is singular to working precision.
ans = 2×1
   Inf
  -Inf
```

3.) Graph

```
ezplot('1 - x',[-10,10])
hold on
ezplot('2*x',[-10,10])
hold on
```

```
ezplot('1-x',[-10,10])
title('Number 3')
grid
hold off
```



Solve

```
A = [1 1 ; 2 -1; -1 -1]
```

```
A = 3×2
     1     1
     2    -1
    -1    -1
```

```
b = [1 ; 0 ; -1]
```

```
b = 3×1
     1
     0
    -1
```

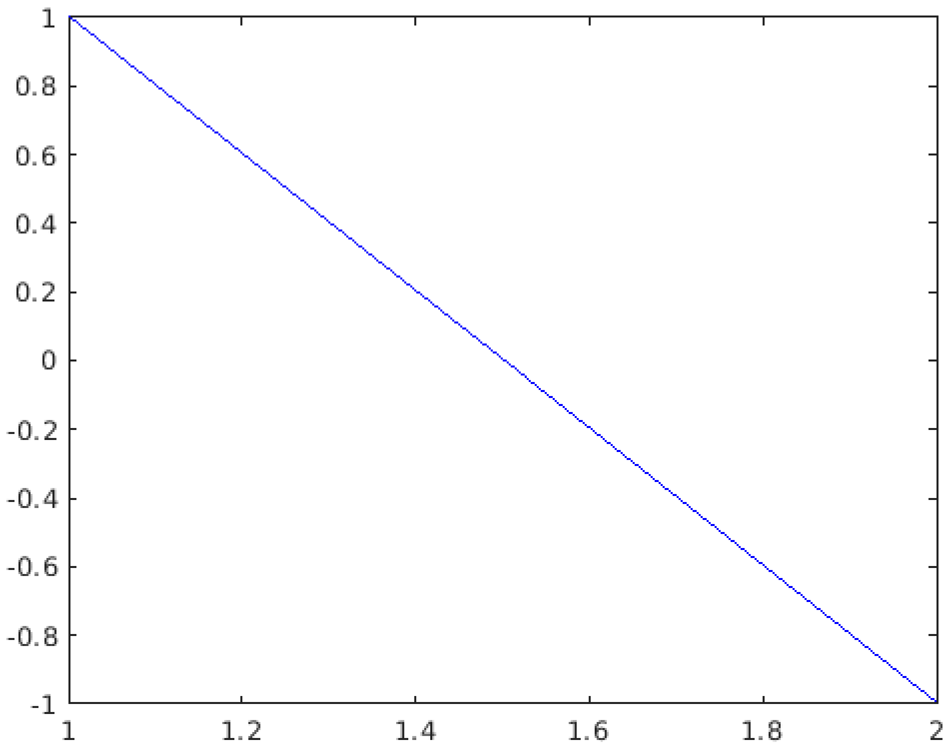
```
A\b
```

```
ans = 2×1
    0.3333
    0.6667
```

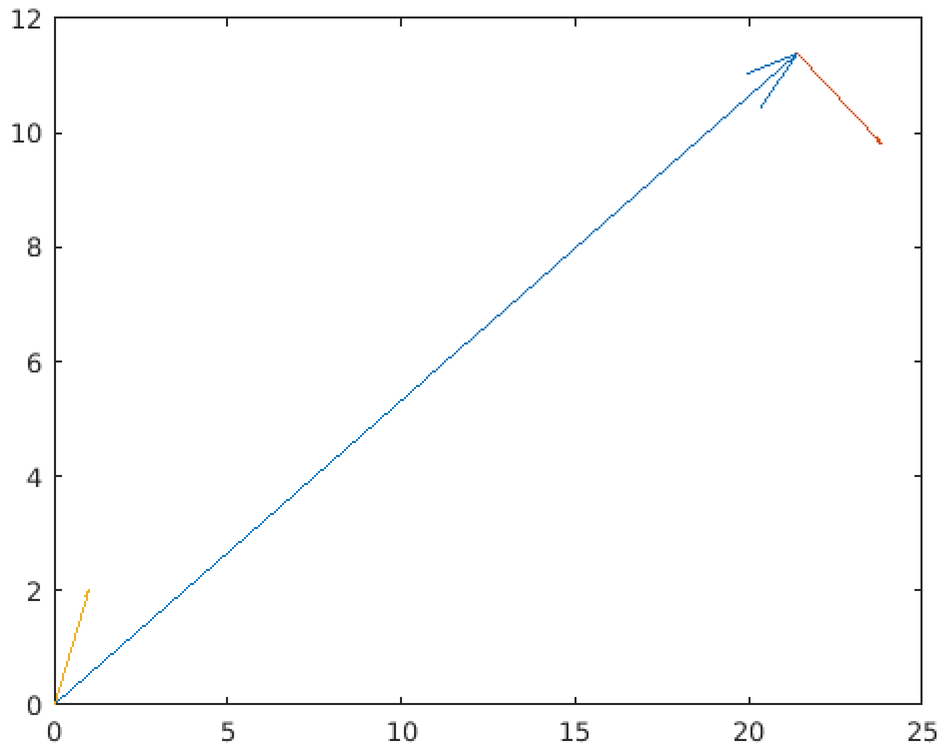
Solution: 1/2 2/3

Column Graph

```
plot([1 2], [1 -1], 'b')
```

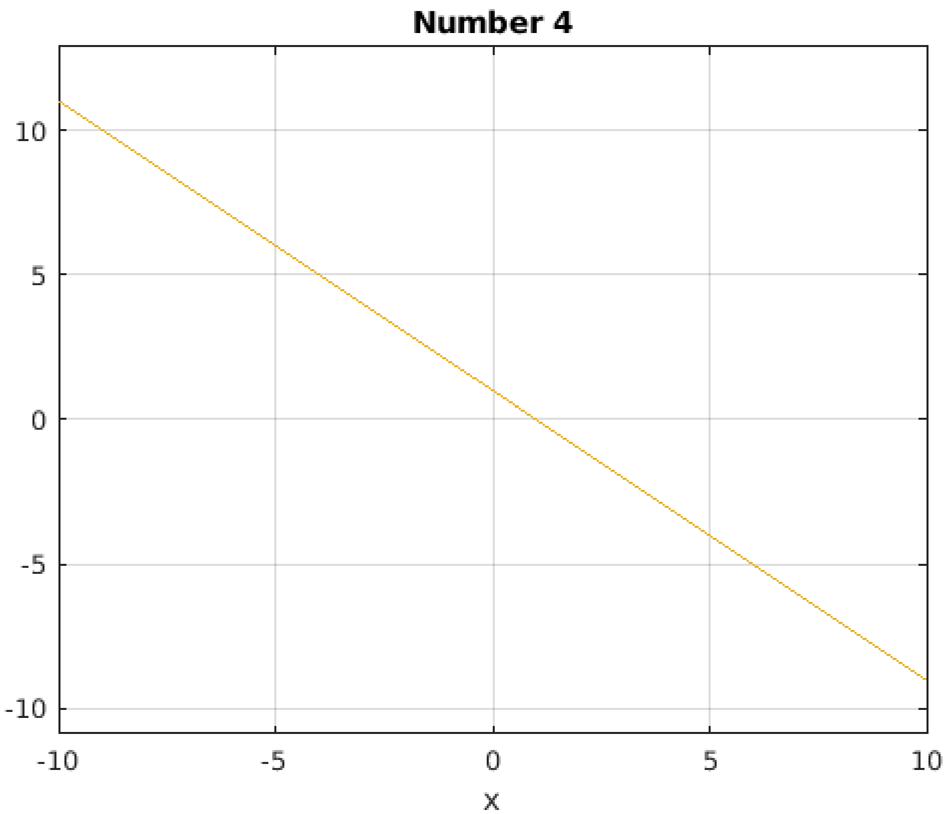



```
quiver(0, 0, 21.4, 11.4, 'AutoScale','off')
hold on
quiver(21.4, 11.4, 3-0.6, -1-0.6, 'AutoScale','off')
quiver(0, 0, 1, 2, 'AutoScale','off')
hold off
```



4.) Graph

```
ezplot('1 - x',[-10,10])
hold on
ezplot('1 - x',[-10,10])
hold on
ezplot('1 - x',[-10,10])
title('Number 4')
grid
hold off
```



Solve

```
A = [1 1 ; -1 -1; 2 2]
```

```
A = 3×2
     1     1
    -1    -1
     2     2
```

```
b = [1 ; -1 ; 2]
```

```
b = 3×1
     1
    -1
     2
```

```
A\b
```

```
Warning: Rank deficient, rank = 1, tol =  1.631688e-15.
ans = 2×1
    1.0000
         0
```

Problem 2.)

```
A = [1 -1 2 ; -3 1 1 ; 1 4 -6]
```

```
A = 3×3
     1    -1     2
```

```
    -3     1     1
     1     4    -6
```

```
B = [0.5 0.35 0.15 ; 0.35 0.6 0.05 ; 0.15 0.05 0.8]
```

```
B = 3×3
    0.5000    0.3500    0.1500
    0.3500    0.6000    0.0500
    0.1500    0.0500    0.8000
```

```
C = [1 -1 ; 1 2 ; -3 2]
```

```
C = 3×2
     1    -1
     1     2
    -3     2
```

1.)

```
2 * 3
```

```
ans = 6
```

2.)

```
A * A
```

```
ans = 3×3
     6     6   -11
    -5     8   -11
   -17   -21    42
```

3.)

```
A * B
```

```
ans = 3×3
    0.4500   -0.1500    1.7000
   -1.0000   -0.4000    0.4000
    1.0000    2.4500   -4.4500
```

4.)

```
B * A
```

```
ans = 3×3
   -0.4000    0.4500    0.4500
   -1.4000    0.4500    1.0000
    0.8000    3.1000   -4.4500
```

5.)

```
A * C
```

```
ans = 3×2
    -6     1
    -5     7
    23    -5
```

6.)

```
%C * A
```

7.)

```
2 * B
```

```
ans = 3×3
    1.0000    0.7000    0.3000
    0.7000    1.2000    0.1000
    0.3000    0.1000    1.6000
```

Did MATLAB refuse to do any of the requested calculations? Why?

Yes, It refused to do 6, because the matrices are not the right dimensions to be multiplied. The number of columns in the first matrix must match the number of rows in the second matrix.

Does A*B = B*A in general?

No, matrix multiplication is not cumulative.

What did 2*B produce? Was that what you expected?

The product was:

1.0000 0.7000 0.3000

0.7000 1.2000 0.1000

0.3000 0.1000 1.6000

It was what I expected because when you multiply two matrices the solution is the dot product of each row by column.

Problem 3.)

```
A = [1 -1 2 ; -3 1 1 ; 1 4 -6]
```

```
A = 3×3
     1    -1     2
    -3     1     1
     1     4    -6
```

```
B = [0.5 0.35 0.15 ; 0.35 0.6 0.05 ; 0.15 0.05 0.8]
```

```
B = 3×3
    0.5000    0.3500    0.1500
    0.3500    0.6000    0.0500
    0.1500    0.0500    0.8000
```

```
C = [1 -1 ; 1 2 ; -3 2]
```

```
C = 3×2
     1    -1
     1     2
    -3     2
```

1.)

```
A'
```

```
ans = 3×3
     1    -3     1
    -1     1     4
     2     1    -6
```

2.)

```
B'
```

```
ans = 3×3
    0.5000    0.3500    0.1500
    0.3500    0.6000    0.0500
    0.1500    0.0500    0.8000
```

3.)

```
C'
```

```
ans = 2×3
     1     1    -3
    -1     2     2
```

4.)

```
C' * A
```

```
ans = 2×3
    -5   -12    21
    -5    11   -12
```

5.)

```
%A * C'
```

6.)

```
(A')'
```

```
ans = 3×3
     1    -1     2
    -3     1     1
     1     4    -6
```

7.)

```
(A' + A)/2
```

```
ans = 3×3
    1.0000   -2.0000    1.5000
   -2.0000    1.0000    2.5000
    1.5000    2.5000   -6.0000
```

Did MATLAB refuse to do any of the requested calculations? Why?

Yes, MATLAB refused to do problem 5. In this case the dimensions of the matrices were not compatible for multiplication. This was because the number of columns in the first matrix must match the number of rows in the second matrix.

Does B = B'? Is B symmetric?

Yes, B = B' in this case. It is symmetric because B = B'. It is equal to its transpose.

```
B = [0.5 0.35 0.15 ; 0.35 0.6 0.05 ; 0.15 0.05 0.8]
```

```
B = 3×3
    0.5000    0.3500    0.1500
    0.3500    0.6000    0.0500
    0.1500    0.0500    0.8000
```

```
B = transpose(B)
```

```
B = 3×3
    0.5000    0.3500    0.1500
    0.3500    0.6000    0.0500
    0.1500    0.0500    0.8000
```

What is the relationship between (A')' and A?

They are equal.

```
(A')'
```

```
ans = 3×3
     1    -1     2
    -3     1     1
     1     4    -6
```

```
A
```

```
A = 3×3
     1    -1     2
    -3     1     1
     1     4    -6
```

Problem 4.)

```
C = [1 1 1 1 ; 1 2 1 2 ; 1 1 1 0 ; 1 4 2 3]
```

```
C = 4×4
     1     1     1     1
     1     2     1     2
     1     1     1     0
     1     4     2     3
```

Numerical Message

“I LOVE MY TA,”

If A = 1, ..., Z = 26

```
Alphabet = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ '
```

```
Alphabet =
```

```
'ABCDEFGHIJKLMNOPQRSTUVWXYZ '
```

```

```

“9 27 12 15 22 5 27 13 25 27 20 1”

```
C*[9;27;12;15]
```

```
ans = 4×1
    63
   105
    48
   186
```

```
C*[22;5;27;13]
```

```
ans = 4×1
    67
    85
    54
   135
```

```
C*[25;27;20;1]
```

```
ans = 4×1
    73
   101
    72
   176
```

Encoding NOON

```
C*[14 ; 15 ; 15 ; 14]
```

```
ans = 4×1
    58
    87
    44
   146
```

Decode Matrix

```
D = inv(C)
```

```
D = 4×4
    -1     2     1    -1
    -2     1     1     0
     3    -3    -1     1
     1     0    -1     0
```

Multiply D by Column Vector

```
a = D*[42;51;34;81]
```

```
a = 4×1
    13
     1
    20
     8
```

```
Alphabet(a)
```

```
ans =
'MATH'
```

Decoding encrypted message

”42 51 34 81,”

```
D = inv(C)
```

```
D = 4×4
    -1     2     1    -1
    -2     1     1     0
     3    -3    -1     1
     1     0    -1     0
```

Decrypt

```
x = D*[42;51;34;81]
```

```
x = 4×1
    13
     1
    20
     8
```

```
Alphabet(x)
```

```
ans =
'MATH'
```

Decypt

”56 83 37 127 42 56 40 83 82 118 55 182 77 119 50 191 48 70 41 121”

```
K = [56 ; 83 ; 37 ; 127]
```

```
K = 4×1
    56
    83
    37
   127
```

```
K1 = D*K
```

```
K1 = 4×1
    20
     8
     9
    19
```

```
K2 = Alphabet(K1)
```

```
K2 =
'THIS'
```

```
P = [42 ; 56 ; 40 ; 83]
```

```
P = 4×1
```

```
    42
    56
    40
    83
```

```
P1 = D*P
```

```
P1 = 4×1
    27
    12
     1
     2
```

```
P2 = Alphabet(P1)
```

```
P2 =
' LAB'
```

```
L = [82 ; 118 ; 55 ; 182]
```

```
L = 4×1
    82
   118
    55
   182
```

```
L1 = D*L
```

```
L1 = 4×1
    27
     9
    19
    27
```

```
L2 = Alphabet(L1)
```

```
L2 =
' IS '
```

```
M = [77 ; 119 ; 50 ; 191]
```

```
M =
    77
   119
    50
   191
```

```
M1  = D*M
```

```
M1 =
    20
    15
    15
    27
```

```
M2 = Alphabet(M1)
```

```
M2 =
'TOO '
```

```
N = [48 ; 70 ; 41 ; 121]
```

```
N =
    48
    70
    41
   121
```

```
N1 = D*N
```

```
N1 =
    12
    15
    14
     7
```

```
N2 = Alphabet(N1)
```

```
N2 =
'LONG'
```

```
strcat(K2,P2,L2,M2,N2)
```

```
ans =
'THIS LAB ISTOOLONG'
```

<u>Explain why you think we must have an invertible matrix to do the encoding?</u>

If a matrix is invertible it means that when a matrix is multiplied by it's inverse it equals the corresponding identitiy matrix. In our case this means that the matrix can be manipulated to serve as a form of encryption/ decryption. We are able to map a vector space into another seperate vector space.

A B = B A = In (Identity Matrix)

The decrypted message is "THIS LAB IS TOO LONG" !