Date No

Linear Algebra (Week 01)

1] System of Sentences

System 1 :-

*] The dog is black
*] The cat is orange
*] The bird is red

Complete [Non-Singular]

System 2 :-

*] The dog is black
*] The dog is black
*] The bird is red

Redundant [Singular]

System 3 :-

*] The dog is black
*] The dog is black
*] The dog is black

Redundant [singular]

System 4 :-

*] The dog is black
*] The dog is white
*] The bird is red

Contradictory [Singular]

ProMate

Date No

Linear Algebra (Week 01)

1] System of Sentences

System 1 :-

*] The dog is black
*] The cat is orange
*] The bird is red

Complete [Non-Singular]

System 2 :-

*] The dog is black
*] The dog is black
*] The bird is red

Redundant [Singular]

System 3 :-

*] The dog is black
*] The dog is black
*] The dog is black

Redundant [singular]

System 4 :-

*] The dog is black
*] The dog is white
*] The bird is red

Contradictory [Singular]

1) System of equations

price of apple = a
price of ~~ap.~~ banana = b

System 1 :-

$$\left.\begin{array}{l} a + b = 10 \\ a + 2b = 12 \\ a = 8,\ b = 2 \text{ (unique)} \end{array}\right\} \text{Complete (Non-Singular)}$$

System 2 :-

$$\left.\begin{array}{l} a + b = 10 \\ 2a + 2b = 20 \\ \left.\begin{array}{l} a = 8, 7, 6 \\ b = 2, 3, 4 \end{array}\right\} \text{(Infinite)} \end{array}\right\} \text{Redundant (singular)}$$

System 3 :-

$$\left.\begin{array}{l} a + b = 10 \\ 2a + 2b = 24 \\ \text{(No solution} \end{array}\right\} \text{Contradictory (singular)}$$

Linear equations	Non-linear
$a + b = 10$	$a^2 + b^2 = 10$
$2a + 3b = 15$	$\sin(a) + b^5 = 15$
$\underbrace{3.4a + 48.99b + 2c}_{\text{Numbers}} = 122.5$	$2^a - 3^b = 0$
	$ab^a + \frac{b}{a} - \frac{3}{b} - \log_{4^a}(c)$

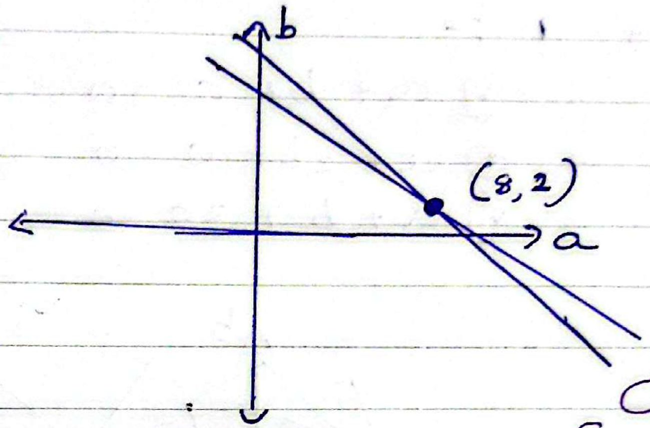
3] System of equations as lines and planes

Lines

1] System 1 :-

$a+b = 10$

$a+2b = 12$



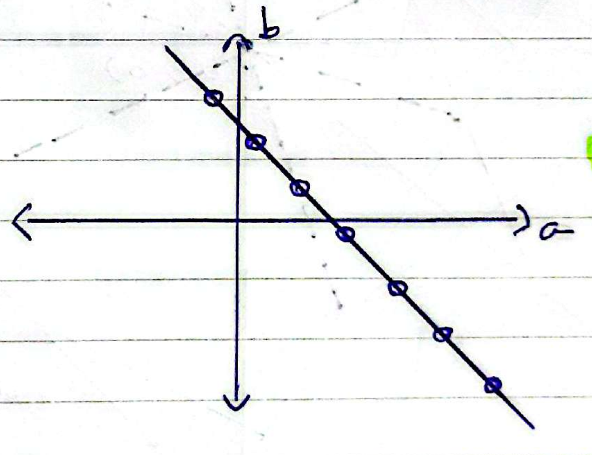
unique Solution

Complete [Non - Singular]

2] System 2 :-

$a+b = 10$

$2a+2b = 20$



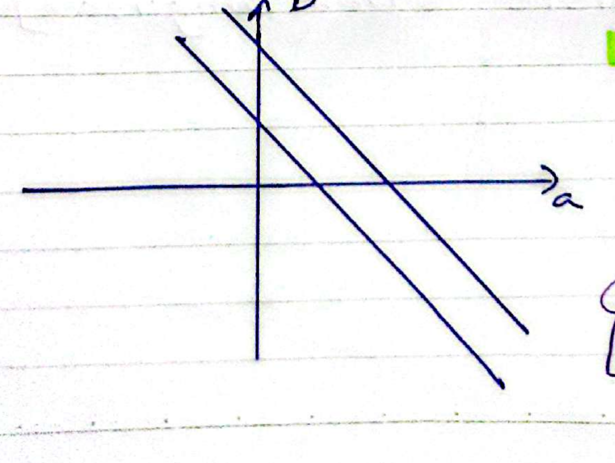
Infinite Solutions

Redundant [Singular]

3] System 3 :-

$a+b = 10$

$2a+2b = 24$



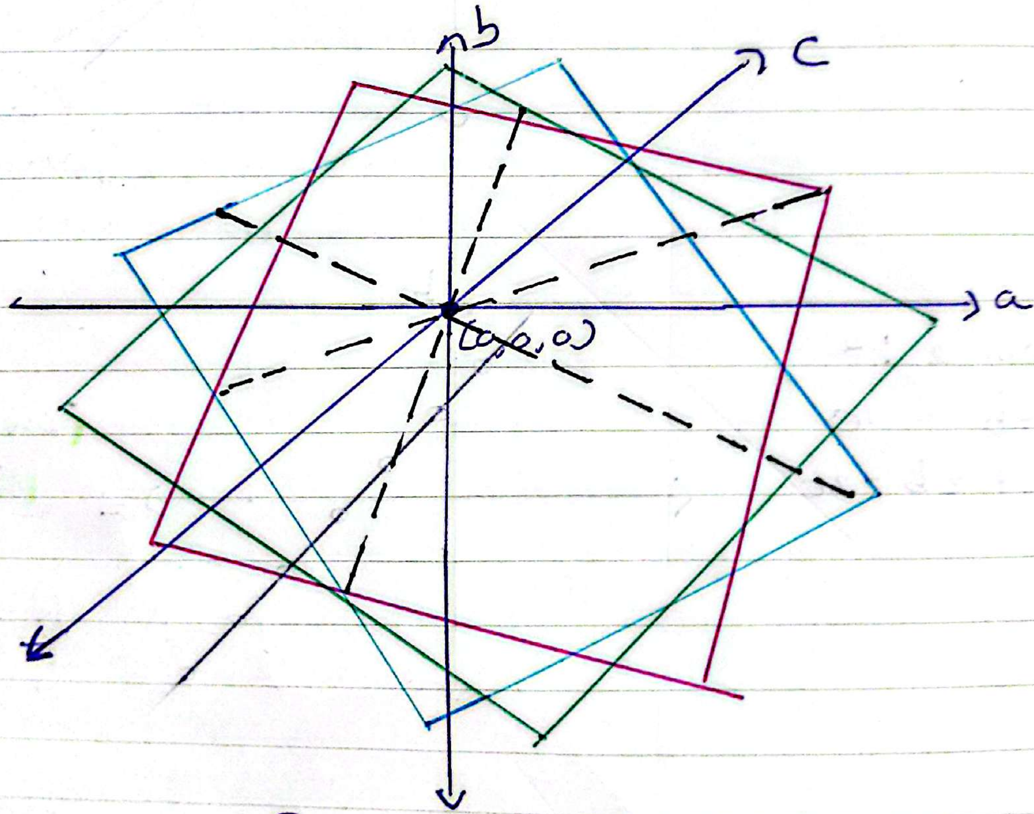
No Solutions

Contradictory [Singular]

Planes

System 1 :-

*) $a + b + c = 0$
*) $a + 2b + c = 0$
*) $a + b + 2c = 0$



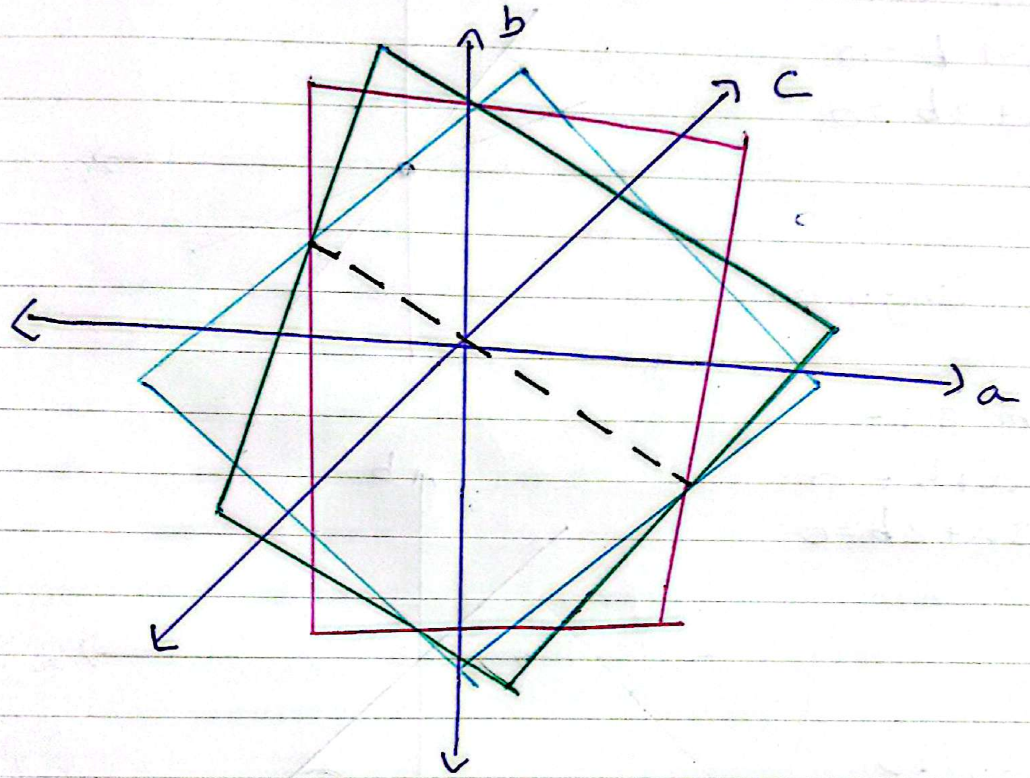
Complete [Non Singular] => Unique Soluti

System 2 :-

$a+b+c=0$

$a+b+2c=0$

$a+b+3c=0$



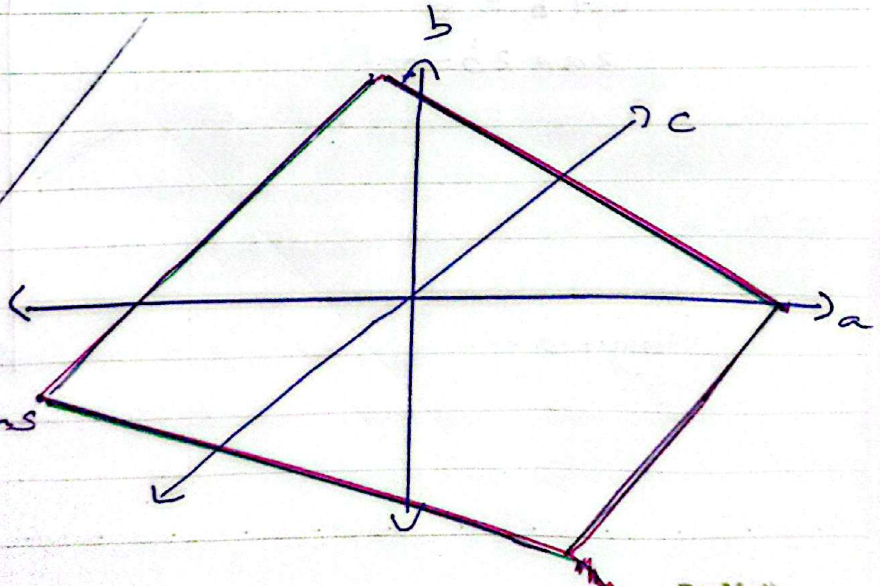
Mutiple solutions [Singular]

System 3 :-

$a+b+c=0$

$2a+2b+2c=0$

$3a+3b+3c=0$



Multiple Soulutions [Singular]

ProMate

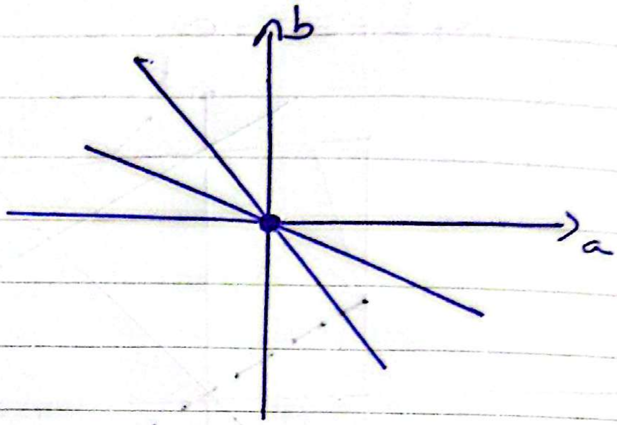
Date

4] A geometric notion of Singularity

i] System 1 :-

$a + b = 0$

$a + 2b = 0$

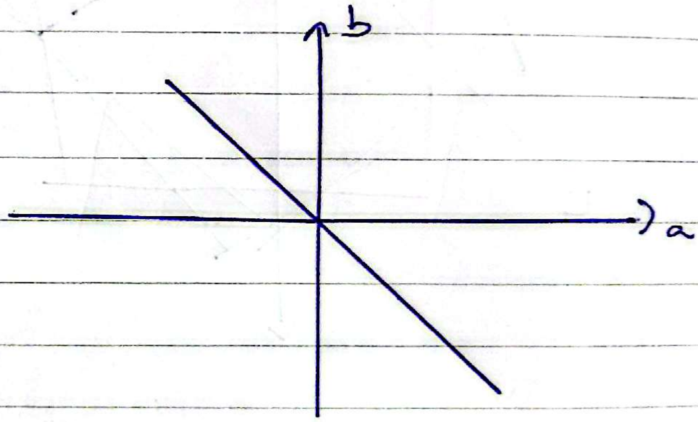


Non Singular

ii] System 2 :-

$a + b = 0$

$2a + 2b = 0$

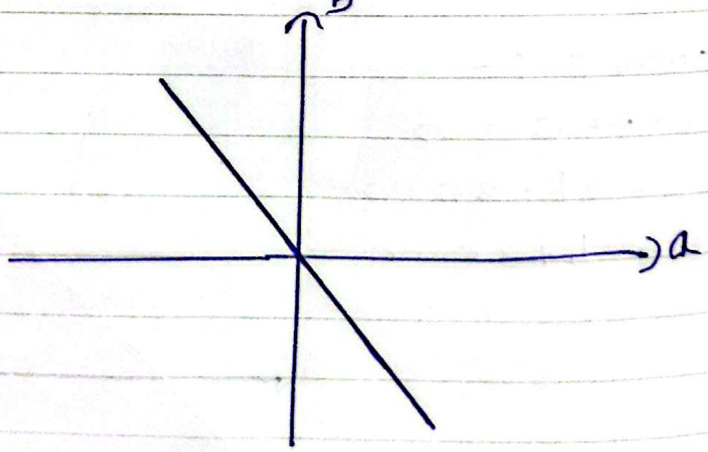


Singular

iii] System 3 :-

$a + b = 0$

$2a + 2b = 0$



Singular

5] Systems of equations as matrices

[*] Constants doesn't matter for singularity

i] System 1 :-

$$\left.\begin{array}{l} a+b+c=0 \\ a+2b+c=0 \\ a+b+2c=0 \end{array} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}\right\}$$

Unique Solution [Non-Singular]

ii] System 2 :-

$$\begin{array}{l} a+b+c=0 \\ a+b+2c=0 \\ a+b+3c=0 \end{array} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 1 & 3 \end{bmatrix}$$

iii] System 3 :-

$$\begin{array}{l} a+b+c=0 \\ a+b+2c=0 \\ a+b+3c=0 \end{array} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 1 & 3 \end{bmatrix}$$

Infinite Solutions [Singular]

iv] System 4 :-

$$\left.\begin{array}{l} a+b+c=0 \\ 2a+2b+2c=0 \\ 3a+3b+3c=0 \end{array} \quad \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix}\right\}$$

Infinite Soluti.. [Singular]

Date

6] Linear dependence and independence

1] $a=1$ $\quad a+0b+0c=1$
$b=2$ $\quad 0a+b+0c=2$
$a+b=3$ $\quad a+b+0c=3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

Row 1 + Row 2 = Row 3
Row 3 depends on Row 1 and Row 2
Rows are linearly dependent

2] $a+b+c=0$
$a+b+2c=0$
$a+b+3c=0$

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

Average of Row 1 and Row 3 is Row 2
Row 2 depends on rows 1 and 3

3] $a+b+c=0$
$a+2b+c=0$
$a+b+2c=0$

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

No relationships between rows
Rows are linearly independent

*] All the rows dont have to be in relationship with each other to be linearly dependent. At least One pair of rows related is enough

ProMate

7] The determinant

$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ Singular if, $a \; b * k = c \; d$

$ak = c$

$bk = d$

$\frac{c}{a} = \frac{d}{b} = k$

$ad = bc$

Determinant → $\boxed{ad - bc} = 0$

*] Singular → Determinant = 0

*] Non-Singular → Determinant $\neq 0$

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

$+$ 4   1 . 2 . 2

$+$ 1   1 . 1 . 1

$+$ 1   1 . 1 . 1

$-$ 2   1 . 2 . 1

$-$ 1   1 . 1 . 1

$-$ 2   1 . 1 . 2

Determinant = 6 − 5
= 1

Shortcut

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 2 & 2 \\ 0 & 0 & 3 \end{bmatrix}$$

*] If 3*3 matrice has all '0's under the main diagnol,

Determinant = 1 . 2 . 3

$1 \times 2 \times 3 = 6$

(Multipiation of the main diaganol)

*] In non-singular matrix, row are linearly independent

*] In singular matrix rows are linearly dependent

Date No

8] Python Matrices & Numpy

Basics of Numpy

*] Numpy is the main package for scientific computing in python. It performs a wide variety of advanced mathematical operations with high efficiency.

1] Packages

*] First we have to import Numpy package to load its functions.

```
import numpy as np
```

2] Advantages of Numpy

*] Numpy provides an array ~~output~~ object that is much faster and more compact than python lists

*] offers many built in functions that make computing much easier with few lines of code.

Date

One_dimensional_arr = np.array([10, 12])
print (one_dimensional_arr)

=) [10, 12]

3) Creating Numpy arrays

# Containing Elements
a = np.array ([1, 2, 3])
print (a)
[1, 2, 3]

3 integers starting from default 0
~~a = np.array~~
a = np.arange (3)
print (a)
[0, 1, 2]

# starts from the integer 1, ends at 20, increment by 3
a = np.arange (1, 20, 3)
print (a)
[1, 4 7 10 13 16 19]

# 5 evenly spaced values
a = np.linspace (0, 100, 5)
print (a)
[0. 25. 50. 75. 100.]

ProMate

data type

```
a = np.linspace(0, 100, 5, dtype=int)
print(a)
 [0  25  50  75  100]

b = np.arange(3, dtype = float)
print(b)
 [0.  1.  2.]

Character = np.array([['welcome to math for ML!']])
print(character)
print(character.dtype)
 ['Welcome to math for ML!]
 <U23          # u = unicode string
               # 23 = number of characters
               # < = endian architecture
```

4) More on Numpy

filled with ones zeroes

```
a = np.ones(3)
b = np.zeroes(3)
print(a)
print(b)
 [1. 1. 1.]
 [0. 0. 0.]
```

Date

\# Without initializing entries
```
a = np.empty (3)
print(a)
[0. 0. 0.]
```

\# Random numbers between 0 and 1
```
a= np.random.rand (3)
print (a)
[0.67584   0.28137   0.99999]
```

Multi-Dimensional Arrays

\# Create 2D Array
```
a = np.array ([[1, 2, 3], [4, 5, 6]])
print (a)
[ [1, 2 3]
  [4 5 6]]
```

```
a = np.array ([1, 2, 3, 4, 5, 6])
# Multi-Dimesion Array using reshape
b= np.reshape (
           a,     # Array to be reshaped
           (2, 3) # dimensions of the new array
           )

print (b)
[[1 2 3
  4 5 6]]
```

ProMate

Date No

Finding size, shape and dimension

Dimension of the array

b. ndim

2

shape of Array (No. of rows x columns)

b. shape

(2, 3)

Size of the array (No. of elements)

b. size

6

Array Math operations

a = np.array([2, 4, 6])
b = np.array([1, 3, 5])

Addition

addition = a + b

print (addition)

[3 7 11]

Subtraction

Substraction = a - b

print (substraction)

[1 1 1]

ProMate

Multiplication

```
multi = a * b
print (mutti)
[ 2  12  30]
```

Multiplying vector with a scalar (broadcasting)

```
Vector = np.array([1, 2])
Vector * 1.6
array ([1.6, 3.2])
```

*) a = np.array ([1, 2, 3])
b = 5
a + b ⟹ [6, 7, 8]
Here 5 is broadcast to [5, 5, 5]

*) A = np. array ([[1, 2, 3],
[4, 5, 6]])

B = np. array ([10, 20, 30])

print (A+B) = [[11 22 33]
[14 25 36]]

Here B is broadcast to $\begin{bmatrix} 10 & 20 & 30 \\ 10 & 20 & 30 \end{bmatrix}$

Date No

Indexing and Slicing

\# Select the 1st element

```
a= ([1, 2, 3, 4, 5]))
print (a[0]))
 1
```

\# indexing on 2d array (i, j)

```
b = np.array ([[ [1, 2, 3],
                 [4, 5, 6],
                 [7, 8, 9] ))
print (b [2][1] )
 8
```

\# get array [2, 3, 4] (slicing)

```
a1 = a[1:4]
print (a1)
 [2, 3, 4]

a2 = a[:3]
print (a2)
 [1, 2, 3]

a3 = a[2:]
print (a3)
 [3, 4, 5]

a4 = a[::2]
print (a4)
 [1, 3, 5]
```

```
print(a == a[:] == a[::])
True
```

Slicing 2D Arrays

```
b1 = b[0:2]
b1
array([[1,2,3],
       [4,5,6]])

b2 = b[1:3]
print(b2)
[[4 5 6]
 [7 8 9]]

b3 = b[:,1]
print(b3)
[2 5 8]
```

Stacking

```
a1 = np.array([[1,1],
               [2,2]])

a2 = np.array([[3,3],
               [4,4]])
print(f'a1:\n {a1}')
print(f'a2:\n {a2}')
```

Date No

a1:

```
[[1, 1]
 [2, 2]]
```

a2:

```
[[3 3]
 [4 4]]
```

stack arrays vertically

```
v = np.vstack((a1, a2))
print(v)
[[1 1]
 [2 2]
 [3 3]
 [4 4]]
```

stack the arrays horizontally

```
h = np.hstack((a1, a2))
print(h)
[[1 1 3 3]
 [2 2 4 4]]
```

→ np.hsplit() - splits an array into several smaller arrays

9] Representing Systems of Equations as Matrices

Representing system of Linear Equations using Matrices

$$-x_1 + 3x_2 = 7$$
$$3x_1 + 2x_2 = 1$$

$$\begin{bmatrix} -1 & 3 & 7 \\ 3 & 2 & 1 \end{bmatrix} = \text{Matrix } A \begin{bmatrix} -1 & 3 \\ 3 & 2 \end{bmatrix}$$

$$\text{Vector } b \Rightarrow \begin{bmatrix} 7 \\ 1 \end{bmatrix}$$

```
import numpy as np
import matplotlib.pyplot as plt
from utils import plot_lines

A = np.array ([
            [-1, 3],
            [3, 2]
            ], dtype= np.dtype (float

b = np.array ([7, 1], dtype = np.dtype.
print ("Matrix A : ")
print (A)
print ("\n Array b : ")
print (b)
```

Date No

Matrix A :

```
[[-1.  -3.]
 [ 3.   2.]]
```

Array b :

```
[7.  1.]
```

```
print (f"Shape of A:  {A.shape}")
print (f"Shape of b : {b.shape}")
Shape of A:  (2, 2)
Shape of b:  (2,)
```

```
# Solve Systems of linear equations
x = np.linalg.solve(A, b)
print (f"Solution: {x}")
```

Solution: [-1. 2.]
x_1 x_2

```
# Evaluate determinant

d= np.linalg.det(A)
print (f"Determinant of Matrix A: {d:.2f}")
```

Determinant of Matrix A : -11.00
(Non-singular)

ProMate

Visualizing 2 × 2 systems as Plotlines

System in the matrix form

```
A = np.hstack((A, b.reshape((2,1))))
print(A)
[[-1.  3.  7.]
 [ 3.  2.  1]]
```

*) Used b.reshape because the current shape of b (2,) is not aligned with the shape of A (2, 2). To do hstack both should be aligned

Graphical Representation

*) A linear equation in two variables (x_1, x_2) Can be represented geometrically by a line in the plane. This is called the graph of the linear equation.

```
plot_lines(A)
```

*) This will give a plot where two lines intersect at (x_1, x_2) = (-1, 2) which is the solution to the system of equations

System of linear equations with No Solutions

$$-x_1 + 3x_2 = 7$$
$$3x_1 - 9x_2 = 1$$

```
A_2 = np.array([
        [-1, 3],
        [3, -9]],
        dtype = np.dtype(float))

b_2 = np.array([7, 1], dtype=np.dtype(float))

d_2 = np.linalg.det(A_2)

print(f"Determinant of Matrix A_2 : {d_2:.2f}")
```

Determinant of Matrix A_2 : 0.00

(Singular, No solutions)

```
# checking singularity
try:
    x_2 = np.linalg.solve(A2, b_2)
except np.linalg.LinAlgError as err:
    print(err)
```

Singular Matrix

Date

plot the singular system

```
A2_system = np.hstack((A_2, b_2.reshape((2,1))))
print (A2_system)
```

```
[ [-1. 3. 7. ]
  [3. -9. 1. ]]
```

```
plot_lines (A2_system)
```

* From this you can see lines of two equations are parallel which means no answers.

System of Linear Equations with an infinite Number of solutions

$$-x_1 + 3x_2 = 7$$
$$3x_1 + -9x_2 = -21$$

```
b3 = np.array([7, -21], dtype=np.dtype(float))

A3_system = np.hstack((A_2, b_3.reshape((2,1))))
print(A3_system)
```

```
[ [ -1. 3. 7.]
  [ 3. -9. -21.]]
```

ProMate

Date No

plot_lines (A3_system)

*) You will see lines of two equations are on each other which means infinite Solutions .