Lab 2

In this lab, you will learn how to:

- write text and math in a text cell,
- use NumPy to do computations, especially with matrices and vectors,
- perform elementary row operations on a matrix,
- use row operations to solve a linear system and related problems,
- experiment with linear transformations and explain their effect on images in the plane.

1 To turn in

Do Problems 1-31 in a single Google Colab notebook. Write *your name* and *lab number* at the beginning of your report. Clearly label each problem to separate them from each other. Make sure to comment on each problem. If your code doesn't run correctly, clearly explain what you were trying to do. Once you are finished, download the .ipynb file by clicking on **File→Download→.ipynb** and download the .pdf file by clicking on **File→Print→Save as PDF**. Submit on Canvas both the *.ipynb file* and the *.pdf file*. Here is the breakdown of points:

Problems	Points
2-4, 6, 11, 13, 16, 23	1
1, 7-9, 12, 15, 18, 24, 25, 28	2
5, 10, 14, 17, 19, 21, 22, 27, 29-31	3
20, 26	4
Readability of your report	3
Total: 31	Total: 72

2 Write text and math in text cell

In Lab 1, you are familiar with writing comments in a text cell. Text cells are processed by a language called *Markdown*.

Type the entire text below in a single *text cell*. If you use Copy and Paste, the line breaks may be skipped. Make sure to add missing line breaks after pasting. Your text cell should have 16 lines.

```
To bold a text, **enclose it within a pair of double stars**.
To italicize a text,  *enclose it within a pair of single stars*.

*In math*, you normally write:
* The **square root** of 2 as $\sqrt{2}$.
* The **fraction** 2 over 5 is written as $\frac{2}{5}$.
* The **sine** of pi over 4 is written as $\sin\frac{\pi}{5}$.
* The **exponential** of 2 is written as $e^2$ or $\exp(2)$.
* The **natural logarithm** of 2 is written as $\ln 2$.

*In Python*, you write the fraction 2 over 5 simply as `2/5`.
To print the output, you write `print(2/5)`.
To compute $\sqrt{2}, \sin\frac{\pi}{5}, e^2, e^{\sqrt{2}}, \ln 2$,
you will need to use a package called **NumPy**.

More details about this package will be provided in the next section.
```

1. What is the role of the pair of double stars `**...**` ? What is the role of the pair of single stars `*...*` ?

2. What is the role of the line breaks between paragraphs (line 3, 10, 15)?

3. What is the role of the stars at the beginning of lines 5-9 ?

4. What is the role of the pair of backticks `` `...` `` on line 11 and 12? (On your keyboard, the backtick is the key below the Esc key.)

5. The text between the pair of dollar signs $...$ is processed by a language call *LaTeX*. LaTeX is a standard language for typesetting math and science in general. Observe how LaTeX works in the above text. Type the following mathematical expression in a *text cell*

$$\frac{\sqrt{3}+e^{3x}}{\ln(x^2+1)}$$

6. If you want to write math in the same paragraph as the text surrounding it (called *inline mode*), use a pair of single dollar signs $...$ as you have seen in the text above. If you want to write math in a standalone line (called *display mode*), use double dollar signs $$...$$ instead. For example, type the following in a *text cell*.

```
The derivative of $\sin(2x)$ is $2\cos(2x)$. For a nicer look, write
$$(\sin(2x))'=2\cos(2x)$$
```

   Replace the pair $$...$$ with $...$ and comment on what you see.

7. To type the matrix

$$A=\begin{bmatrix} a_1 & a_2 \\ 5 & 6 \\ v_1 & v_2 \end{bmatrix}$$

   type the following in a *text cell*.

```
$$
A=\begin{bmatrix}
a_1 & a_2\\
5 & 6 \\
v_1 & v_2
\end{bmatrix}
$$
```

   From the above example, can you guess the roles of symbol & and the symbol \\? Now type the following matrix in a *text cell*.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

8. To type the vector

$$v_1=\begin{bmatrix} 2 \\ 0 \\ -1 \end{bmatrix}$$

   type the following in a *text cell*.

```
$$
v_1=\begin{bmatrix}
2 \\
0 \\
-1
\end{bmatrix}
$$
```

Now type the following vector equation in a *text cell*.

$$x_1 \begin{bmatrix} 1 \\ 2 \end{bmatrix} + x_2 \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$

3 NumPy

Although there are a number of useful functions which are already defined in Python, such as **print**, **range**, **len** functions, there are many common mathematical functions like sine of x and logarithm of x, which are not defined. In order to use these functions, you need to import the NumPy package. In Python, a package is a collection of functions that have been written in Python and are available to use so that you don't have to define these functions yourself. NumPy is a particularly helpful package that contains many functions which are important for doing linear algebra and mathematics in general. In order to use the functions in the NumPy package, you first must import the package by executing the following command (in a *code cell*):

```
import numpy as np
```

Here, you are telling Python to import the NumPy package and referring to it later by the abbreviation `np`, instead of by its full name. (Of course, you can choose a different abbreviation, but `np` is quite commonly used and which we will use in this lab.) You need to do this for every notebook you create that uses NumPy's functions. If you close the notebook which has imported NumPy and then open it again, you will need to re-execute the command mentioned above in order to use any of NumPy's functions again. To use NumPy's functions in your code, you simply have to include "`np.`" at the beginning of the function name.

9. For example, if you want to compute sine of 1/2, execute the following in a *code cell*:

   ```
   np.sin(0.5)
   ```

   The command for the square root of 2 is `np.sqrt(2)`. The command for the natural logarithm of 2 is `np.log(2)`. The command for the exponential of 2 is `np.exp(2)` or equivalently, `np.e**2`. In a single *code cell*, compute the following quantities: $\sqrt{2}$, $\ln 3$, e^4, $\cos(5)$. Remember to use the **print** command to see multiple outputs. Write your comment in a *text cell*.

10. Write a code to compute

    $$\frac{e^5 - \ln(\sqrt{5})}{e^{\cos(3)} + 5^5}$$

    Then write your comment in a *text cell*.

Working with vectors:

Another useful feature of the NumPy package is that it also contains functions for dealing with vectors and matrices. In NumPy, we represent matrices and vectors as *arrays*. To define a NumPy array, we use the function `np.array`. For example, if we want to create the vector

$$\begin{bmatrix} 2 \\ 0 \\ -1 \end{bmatrix}$$

as a NumPy array, we apply the function `np.array` to the list `[1,2,-1]`.

11. Execute the following code:

```
v = np.array([1,2,-1])
print(v)
```

12. Note that NumPy arrays behaves differently compared to Python lists. In linear algebra, we usually want vectors to behave like NumPy arrays, instead of Python lists. For that reason, we will use NumPy arrays to represent vectors and matrices instead of lists. For instance, execute the following in a *code cell*:

```
list1 = [1,2,3]
list2 = [2,-1,0]
print(list1+list2)
```

    Then execute the following:

```
array1=np.array(list1)
array2=np.array(list2)
print(array1+array2)
```

    How does Python combine the lists `list1` and `list2` when we use the command `list1+list2`? How does Python combine the NumPy arrays `array1` and `array2` when we use the command `array1+array2`? Comment on what you see in a *text cell*.

Working with matrices:

To define a matrix in NumPy, we define them as a "list of lists".

13. For example, to define the matrix

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ -5 & -6 & -7 & -8 \\ 1 & 5 & 2 & 3 \end{bmatrix}$$

    you will execute the following

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

14. You can access elements of a NumPy array the same way you access elements in a list, by specifying indices or ranges of indices. Execute the following:

```
v = np.array([4,1,-5,3,-2,1,0,9])
A = np.array([[1, 2, 3, 4],[-5, -6, -7, -8],[1, 5 , 2, 3]])
print(v[3])
print(v[2:6])
print(v[3:])
print(v[:6])
print(v[:])
print(A[1,2])
print(A[2,1:3])
print(A[:,3])
print(A[2,:])
```

Comment on what each command does.

4 Solve linear systems

Consider the linear system

$$\begin{cases} x + y + z = 2 \\ 2x - y - 2z = 5 \\ 3x + z = 5 \end{cases} \tag{1}$$

whose augmented matrix is

$$\begin{bmatrix} 1 & 1 & 1 & 2 \\ 2 & -1 & -2 & 5 \\ 3 & 0 & 1 & 5 \end{bmatrix}$$

The *Gauss-Jordan elimination method* is the method of solving a linear system by performing row operations on the augmented matrix until it reaches the reduced echelon form. The first two row operations are $R_2 = R_2 - 2R_1$ and $R_3 = R_3 - 3R_1$, which make the first column a pivot column.

$$\begin{bmatrix} 1 & 1 & 1 & 2 \\ 2 & -1 & -2 & 5 \\ 3 & 0 & 1 & 5 \end{bmatrix} \xrightarrow[R_3=R_3-3R_1]{R_2=R_2-2R_1} \begin{bmatrix} 1 & 1 & 1 & 2 \\ 0 & -3 & -4 & 1 \\ 0 & -3 & -2 & -1 \end{bmatrix}$$

15. Enter the augmented matrix as a NumPy array. Name it A. Print out the matrix.

16. To perform the row operation $R_2 = R_2 - 2R_1$, execute the following

    ```
    A[1,:] = A[1,:] - 2*A[0,:]
    print(A)
    ```

    Note that A has already been changed. If you accidentally make a typo mistake in the row operation, you will have to re-execute the definition of A to restore its original form and do the row operation again.

17. Do the remaining row operations to turn matrix A into reduced echelon form. Print out the matrix at each step.

18. Write the solution to the linear system (1) in a *text cell*.

19. Solve the linear system

$$\begin{cases} x_1 + 2x_2 + x_4 = 0 \\ x_3 - 2x_4 = 3 \end{cases} \tag{2}$$

    Write the solution in parametric vector form in a *text cell*.

20. Balance the following chemical equation

$$KMnO_4 + HCl \to KCl + MnCl_2 + H_2O + Cl_2$$

In other words, find positive intergers $x_1, x_2, ..., x_6$ such that

$$x_1KMnO_4 + x_2HCl = x_3KCl + x_4MnCl_2 + x_5H_2O + x_6Cl_2$$

You will need to write down the linear system so that the number of atoms of each chemical element (K, Mn, O, H, Cl) on both sides are equal to each other. If the system has infinitely many solutions, just choose one in which $x_1, x_2, ..., x_6$ are all integers.

5 Linear combination and linear independence

A lot of problems in linear algebra can be turned into a problem of solving a linear system. Let us consider two types of problems below.

Linear combination

How do we check if a vector v if a linear combination of a set of vectors $v_1, v_2, ..., v_n$? We will need to solve the vector equation

$$x_1v_1 + x_2v_2 + \ldots + x_nv_n = v.$$

This is equivalent to solving a linear system equation whose unknowns are $x_1, x_2, \ldots, x_n$ and whose augmented matrix is obtained by patching together vectors the $v_1, v_2, ..., v_n, v$ as columns.

$$\begin{bmatrix} | & | & \ldots & | & | \\ v_1 & v_2 & \ldots & v_n & v \\ | & | & \ldots & | & | \end{bmatrix}$$

Then you use Gauss-Jordan elimination (row operation) method to solve the system. If the system is consistent (meaning, there is a solution) then v is a linear combination of $v_1, ..., v_n$. Otherwise, v is not.

Linear independence

How do we check if a set of vectors $v_1, v_2, ..., v_n$ is linearly independent? We will solve the vector equation

$$x_1v_1 + x_2v_2 + \ldots + x_nv_n = 0$$

to see if there are any solutions other than the trivial solution $x_1 = x_2 = \ldots = x_n = 0$.

21. Check if $v = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$ is a linear combination of $v_1 = \begin{bmatrix} -1 \\ 0 \\ 3 \end{bmatrix}$ and $v_2 = \begin{bmatrix} 2 \\ 1 \\ -2 \end{bmatrix}$.

22. Check if the vectors $v_1 = \begin{bmatrix} -1 \\ 0 \\ 3 \end{bmatrix}$, $v_2 = \begin{bmatrix} 2 \\ 1 \\ -2 \end{bmatrix}$, $v_3 = \begin{bmatrix} 4 \\ 3 \\ 0 \end{bmatrix}$ are linearly independent. If they are not, find a *linear dependence relation* among them. That is a relation of the form $x_1v_1 + x_2v_2 + x_3v_3 = 0$ in which x_1, x_2, x_3 are not all zeros.

6 Linear transformation

Consider a linear transformation $T : \mathbb{R}^2 \to \mathbb{R}^2$ given by

$$T\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} 2x + y \\ x - 3y \end{bmatrix} \tag{3}$$

It is represented by a 2×2 matrix

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

in the sense that for all vectors $v \in \mathbb{R}^2$, we have $T(v) = Av$. You obtain matrix A in one of the following ways:

- The first column of A lists the coefficients of x. The second column lists the coefficients of y.
- You can also see that the first column of A is $T(e_1)$, and the second column is $T(e_2)$ where $e_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $e_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ are the *standard basis vectors*. (For this reason, a linear transformation T is completely determined if you know its value at the standard basis vectors.)

23. To multiply two NumPy matrices, you use the operator `@`. For example, execute the following

```
A = np.array([[1, 2, 3, 4], [-5, -6, -7, -8], [1, 5, 2, 3]])
v = np.array([0, 1, -1, 0])
print(A@v)
```

24. Define a function named `transform` which takes as input a NumPy vector $v \in \mathbb{R}^2$ and returns the vector $T(v)$ resulting from (3) as a NumPy vector. Test your function with a few vectors. For example, `transform([1,2])` should return `[4,-5]`.

Geometric interpretation of linear transformation

For illustration purposes, we use the vector notation $\begin{bmatrix} x \\ y \end{bmatrix}$ interchangeably with the notation (x, y), which represents a point on the coordinate plane. You can interpret the linear transformation T given by (3) as a function that takes as input a point (x, y) on the plane and returns another point $(2x + y, x - 3y)$, called the *image of* (x, y) *under the linear transformation* T.

Suppose that you have a list of points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ and you want to compute the images of those points, namely $T(x_1, y_1)$, $T(x_2, y_2)$,..., $T(x_n, y_n)$. Let us write $v_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$, $v_2 = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$,..., $v_n = \begin{bmatrix} x_n \\ y_n \end{bmatrix}$. We know that

$$T(v_1) = Av_1, \quad T(v_2) = Av_2, \ \ldots \quad T(v_n) = Av_n.$$

We can combine these vector equations into a matrix equation by attaching the vectors into a matrix:

$$[T(v_1) \ \ T(v_2) \ \ \ldots \ T(v_n)] = [Av_1 \ \ Av_2 \ \ \ldots \ Av_n].$$

The right hand side is known as the *multiplication* of matrix A and matrix $[v_1 \ \ v_2 \ \ \ldots \ v_n]$, denoted by $A[v_1 \ \ v_2 \ \ \ldots \ v_n]$. Thus, we have

$$[T(v_1) \ \ T(v_2) \ \ \ldots \ T(v_n)] = A[v_1 \ \ v_2 \ \ \ldots \ v_n].$$

25. For example, to find the images of three points $(1,2)$, $(0,1)$, $(2,1)$ under the transformation T, we write

$$[T(v_1)\ \ T(v_2)\ \ T(v_3)] = A[v_1\ \ v_2\ \ v_3] = \begin{bmatrix} 2 & 1 \\ 1 & -3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 \\ 2 & 1 & 1 \end{bmatrix}$$

Execute the following in a *code cell*.

```
A=np.array([[2,1],[1,-3]])
B=np.array([[1,0,2],[2,1,1]])
print(A@B)
```

From the output, can you tell the images of the points $(1,2)$, $(0,1)$, $(2,1)$ under the linear transformation T ?

26. Let $T : \mathbb{R}^2 \to \mathbb{R}^2$ be a linear transformation such as $T(e_1) = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ and $T(e_2) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$.

- Enter the matrix of this linear transformation. Make sure to print it out.
- Find the images of the points $(1,1)$, $(2,3)$, $(-1,0)$, $(3,-1)$ under the transformation.

27. The function `show` defined below takes as input a matrix H of size $2 \times n$, consisting of the coordinates of n points $(x_1, y_1), \ldots, (x_n, y_n)$:

$$H = \begin{bmatrix} x_1 & x_2 & \ldots & x_n \\ y_1 & y_2 & \ldots & y_n \end{bmatrix}$$

and plot these points on the coordinate plane. Execute the definition of function `show`:

```
from matplotlib import pyplot as plt
def show(H):
    plot=plt.plot(H[0,:],H[1,:],'k.',markersize=3.5)
    plt.axis([-1.5,1.5,-1.5,1.5])
    plt.gca().set_aspect("equal")
    plt.show()
```

Consider the matrix

$$H = \begin{bmatrix} 0.5 & -0.5 & 1 & -0.7 \\ -0.9 & 0.6 & 0 & 0.5 \end{bmatrix}$$

- Use function `show` to plot the 4 points whose coordinates are columns of matrix H.
- Consider a linear transformation whose matrix is $A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. Use function `show` to plot the images of the 4 points under this linear transformation.

Experiments with the L shape

The file `Lshape.csv` contains the coordinate of 10000 points whose plot forms an L shape. The simplest way to load the data into Google Colab is to first download it (397 kB) to your local computer by clicking the link:

https://drive.google.com/uc?export=download&id=1IV6mr6Q04Swng2J-RkxkU_V2MSoUWU8-

On the left sidebar of Google Colab, you should see a little Folder icon. Selecting it opens a new window to the left of the notebook. Click on "Upload" (the icon with an arrow pointing up). This should bring up a window that allows you to select the file "Lshape.csv" from your local machine, which will upload the file to your notebook. You will need to do this again if you close your notebook and reopen it at a later time.

28. Execute the following code to convert the dataset in the file `Lshape.csv` into a NumPy matrix called L.

```
import pandas as pd
from matplotlib import pyplot as plt
file = pd.read_csv('Lshape.csv')
L=file.values.T
print(L)
```

Execute `L.shape` to see the size of matrix L. How many rows and columns does it have?

29. A **stretch** is a linear transformation which stretches or compresses a shape along the horizontal and/or vertical axes. The matrix representation of a stretch is

$$A = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$

This linear transformation stretches a shape horizontally by factor a and vertically by factor b. Define a function `stretch(H,a,b)` that accepts as input a $2 \times n$ array H and parameters a and b needed to define the stretch. The function should return the array which is obtained from set of points by applying the stretch. Then run two tests of your function: `stretch(L,1.5,1.5)` and `stretch(L,0.5,1.2)`. For each test, plot the new shape of the letter L using the function `show` already defined in Exercise (27).

30. A **rotation** is a linear transformation which rotates a shape around the origin. A counterclockwise rotation of θ radians has the following matrix representation

$$A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Define a function `rotate(H,theta)` that accepts as input a $2 \times n$ array H and parameter θ needed to define the rotation. The function should return the array which is obtained by applying the rotation to the set of points. Run two test by applying your function to the shape L: one with $\theta = 2\pi/3$ and one with $\theta = -2\pi/3$. For each test, plot the new shape of the letter L.

31. Plot the shape of letter L if you perform the following sequence of transformations (in order):

- Stretch horizontally by a factor of 2.
- Rotate $\pi/4$ radians clockwise.
- Stretch vertically by a factor of $1/2$.