Adamson University
College of Engineering
Computer Engineering Department

Linear Algebra

Laboratory Activity No. 2

# Plotting Vectors using NumPy and MatPlotLib

*Submitted by:*

Roa, Danica Kate I.

*Instructor:*

Engr. Dylan Josh D. Lopez

October 04, 2020

# I. Objectives

This laboratory activity aims to implement the principles and techniques of Plotting Vectors using NumPy and MatPlotLib.

# II. Methods

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. *(https://en.wikipedia.org/wiki/NumPy)*

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.( *[https://en.wikipedia.org/wiki/Matplotlib](https://en.wikipedia.org/wiki/Matplotlib))*

# III. Results

```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib

%matplotlib inline
def track_eagle(make_figs=True):
    long = np.random.randint(-10,10, size=5)
    lat =  np.random.randint(-10,10, size=5)


    dist1 = np.array([long[0],lat[0]])
    dist2 = np.array([long[1],lat[1]])
    dist3 = np.array([long[2],lat[2]])

    dist_total = dist1 + dist2 + dist3
    disp  = np.linalg.norm(dist_total)
    alpha = 10**-6
    theta = np.arctan((dist_total[1])/(dist_total[0] + alpha))
    theta = np.degrees(theta)

    ## Plotting the PH Eagle flight vectors.
    plt.figure(figsize=(10,10))
    plt.title('Philippine Eagle Flight Plotter')
    plt.xlim(-30, 30)
    plt.ylim(-30, 30)
    plt.xlabel('Latitudinal Distance')
    plt.ylabel('Longitudinal Distance')
    plt.grid()
    n = 2

    plt.quiver(0,0, dist1[0], dist1[1],
               angles='xy', scale_units='xy',scale=1, color='lime',
               label='Trajectory 1: {:.2f}m.'.format(np.linalg.norm(dist1)))

    plt.quiver(dist1[0], dist1[1], dist2[0], dist2[1],
               angles='xy', scale_units='xy',scale=1, color='aqua',
              label='Trajectory 2: {:.2f}m.'.format(np.linalg.norm(dist2)))

    plt.quiver(np.add(dist1[0],dist2[0]), np.add(dist1[1],dist2[1]),
               dist3[0], dist3[1], angles='xy', scale_units='xy',scale=1, color='purple',
              label='Trajectory 3: {:.2f}m.'.format(np.linalg.norm(dist3)))

    plt.quiver(0,0, dist_total[0], dist_total[1],
               angles='xy', scale_units='xy',scale=1, color='maroon',
              label='Displacement: {:.2f}m. @ {:.2f}'.format(disp, theta))

    plt.legend()


    if make_figs:
        plt.savefig(f'LinAlg-Lab2-PH Eagle-{int(disp)}@{int(theta)}.png', dpi=300)

    plt.show()

track_eagle(make_figs=False)
```

Figure 1

It is where we assign specific value and the color of each arrow in the program and also, it is where we construct the size of the graphing area for our input.

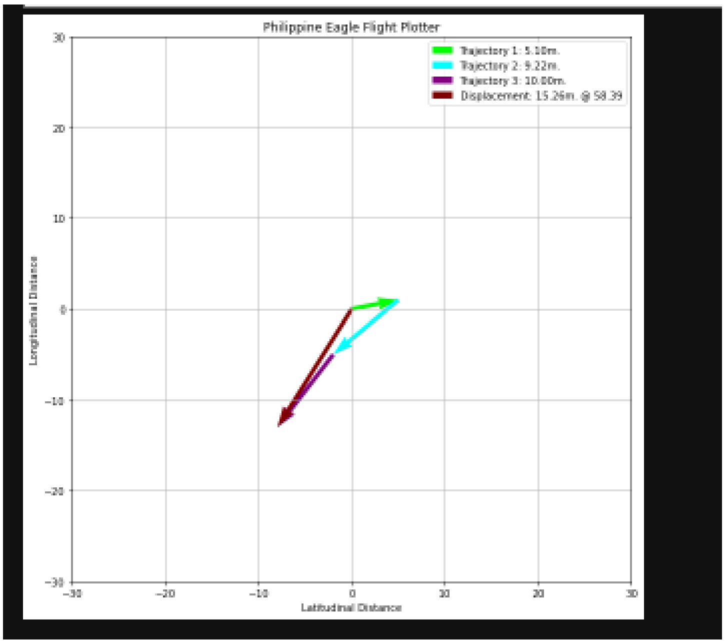


Figure 2

The result of the given data that was plotted in a specific measurement and their colors corresponds to each variable.

```
def eagle_kinematics(position, time):
    req_shape = 4
    velocity = np.zeros((req_shape-1,))
    acceleration = np.zeros((req_shape-2,))
    total_vector = np.array([t**3, t**2, t, 1])
    if position.shape == (req_shape,):
        velocity = np.array([3*position[0],2*position[1], position[2]])
        acceleration = np.array([2*velocity[0],velocity[1]])
        position_total = np.sum(np.multiply(position, total_vector))
        velocity_total = np.sum(np.multiply(velocity, total_vector[1:]))
        acceleration_total = np.sum(np.multiply(acceleration, total_vector[2:]))
    else:
        print(f'Input displacement vector is not valid. Make sure that the vector shape is ({req_shape},)')

    return position_total, velocity_total, acceleration_total

x = np.array([2,1,3,2])
t = 2
eagle_kinematics(x, t)
```

```
(28, 31, 26)
```

Figure 3

This is the part when you can put array to have specific answers

```python
def month_profit_trace(profit, reach, make_figs=True):
    if (profit.shape == (4,)) and (reach.shape == (4,)):
        week1 = np.array((reach[0], profit[0]))
        week2 = np.array((reach[1], profit[1]))
        week3 = np.array((reach[2], profit[2]))
        week4 = np.array((reach[3], profit[3]))

        week_total = week1 + week2 + week3 + week4
        week_performance = np.linalg.norm(week_total)
        alpha = 10**-6
        reach_gradient = np.arctan((week_total[1])/(week_total[0] + alpha))
        reach_gradient = np.degrees(reach_gradient)


        plt.figure(figsize=(16,5))
        plt.title('Bebang\'s Month Post Efficiency')
        plt.xlim(0,1.01*np.sum(reach))
        plt.ylim(-np.sum(np.abs(profit)),np.sum(np.abs(profit)))
        plt.xlabel('FB Post Reach Increment')
        plt.ylabel('Profit')
        plt.grid()


        plt.quiver(0,0, week1[0], week1[1],
                   angles='xy', scale_units='xy',scale=1, color='green', width=0.0025,
                   label='Week 1: {:.2f}'.format(np.linalg.norm(week1)))

        plt.quiver(week1[0], week1[1], week2[0], week2[1],
                   angles='xy', scale_units='xy',scale=1, color='yellow', width=0.0025,
                   label='Week 2: {:.2f}'.format(np.linalg.norm(week2)))

        plt.quiver((week1[0] + week2[0]), (week1[1] + week2[1]), week3[0], week3[1],
                   angles='xy', scale_units='xy',scale=1, color='aqua', width=0.0025,
                   label='Week 3: {:.2f}'.format(np.linalg.norm(week3)))

        plt.quiver((week1[0] + week2[0] + week3[0]), (week1[1] + week2[1] + week3[1]), week4[0], week4[1],
                   angles='xy', scale_units='xy',scale=1, color='maroon', width=0.0025,
                   label='Week 4: {:.2f}'.format(np.linalg.norm(week4)))


        plt.quiver(0,0, week_total[0], week_total[1],
                   angles='xy', scale_units='xy',scale=1, color='purple', width=0.005,
                   label='Efficiency: {:.2f} @ {:.2f}'.format(week_performance, reach_gradient))

        plt.legend(loc='upper left')

        if make_figs:
            plt.savefig(f'LinAlg-Lab2-Bebang Post Eff-{int(week_performance)}@{int(reach_gradient)}.png', dpi=300)

        plt.show()

    else:
        print('Wait for the month to finish to calculation of the Monthly Post Efficiency')


profit= np.array([-18000, 3000, 12000, 10000])
reach = np.array([1000, 100, 500, 10])

month_profit_trace(profit, reach, make_figs=False)
```

Figure 4

This is where you assign specific value that were asignt o the problem and also this is where creates an array for the weekly with reach and profit as the elements

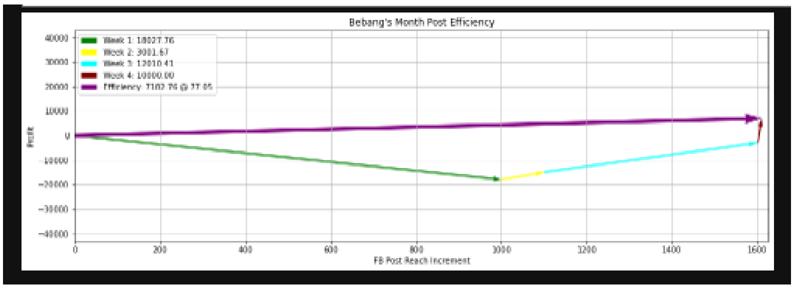


Figure 5

The final output of number four (4) which is already plotted in the graphing area.

## Discussions

1. Enumerate and briefly discuss the functions you have used in the laboratory exercise, please cite their usage using their respective documentations. (min of 200 words)

The function random which will assign random integer automatically in the activity, Array which help to identify each variable in the vectors, theta which help to convert a specific value to what you need, the parameter which will help to identify the x and y values and the direction of the arrows, plt.quiver is use to plot the arrows and you can assign specific color so you can easily identify the arrow fast. plt. Legend will show what are the corresponding colors and it will help the view to see fast what the need to seen. Plt show is like a print function which automatically display the outcome. Using if else statement to identify what should be the outcome of the ] inputted variable or integers. 'make_figs' during debugging or making documentations and it makes your work easier. This is just the basic function but it is very helpful to us beginners it is where we can learn and it can make our foundation to programming more strong.

2. How do vectors relate to real-life values? (min of 50 words)

Vectors are related to real-life through sailing a big boat and navigating in aircraft. This may be not as other think but the way how the captain of the boat and the pilot is through dimensions in longitude and latitude in the air and in water to secure the trip

3. Kindly give other examples of how vectors are used or other real-life situations that can be modeled using vectors? (min of 100 words and do proper citation)

The billiard ball's momentum is also a vectorial quantity, because momentum is equal to mass times velocity. Therefore, the ball's momentum vector points in the same direction as its velocity vector, and the momentum vector's magnitude, or length, is the multiplication product of the ball's speed and its mass. *( https://www.scientificamerican.com/article/football-vectors/)*

Figuring out the direction of rain and holding your umbrella in that direction. To move an object in a particular direction, we will have to apply requisite force in that specific direction. *( https://www.freeaptitudecamp.com/applications-of-vectors-in-real-life/)*

# IV. Conclusion

I conclude that knowing what are the real use of each function will help you a lot interms of activities. Not only in school but in future. It may be simple as of now but in the future it will be the foundation of our knowledge and with the help of this course.

# References


[1] D.J.D. Lopez. "Adamson University Computer Engineering Department Honor Code," AdU-CpE Departmental Policies, 2020.

[2] *https://www.scientificamerican.com/article/football-vectors/*

*[3] https://www.freeaptitudecamp.com/applications-of-vectors-in-real-life/*