Adamson University
College of Engineering
Computer Engineering Department

Linear Algebra

Laboratory Activity No. 2

Plotting Vectors using NumPy and MatPlotLib

Submitted by:
Columba, Lorenzo Miguel L.

Instructor:
Engr. Dylan Josh D. Lopez

October 10, 2020

I. Objectives

In this laboratory activity aims to implement the essential principles in using matplotlob.py and numpy and to familiarize with the libraries in Python for numerical and scientific programming and to examine the vectors, its plotting with its operation.

II. Methods

In the practices of activity are used to implement the function of importing NumPy and matplotlib to indicate its specific usage corresponding to plt.quiver(), np., plt.figures(), plt.x and y lim, plt.grid(), plt.legends(), shape, and size. The deliverables of this activity are to show the plotting of the eagle's flight and therefore the on-line business of Bebang. For the eagle kinematics, is to supply correct documentation of the codes so as to know the method of the code. so as to realize them, I used totally different functions and objects within the drawback and supplemented every missing code from its problem. Also, to understand the process of distribution and career its variables and function. It took me every week to finalized its missing code and correct it properly.

III. Results

The first problem is to identify and show the trajectory of the eagle's flight. In order to get the values of longitude and latitude is to assign its random integers using the np.randomint(). Then I assigned the arrays starting from zero to two. Then I used the formula of Pythagorean's theorem using np.sqrt() to get the magnitude displacement of the vectors. I used the plt.quiver() to represent velocity vectors of the trajectory by arrows. There 4 quivers in the problem, the first trajectory represented by the red arrow, the second trajectory is the arrow blue, the third trajectory is the green arrow and for the entire displacement is that the orange arrow. The 3 displacements are connected to every different and it's all added along to get the coordinates of the total displacement of the eagle's flight, "theta = np.arctan(dist_total[1]/(alpha+ dist_total[0]))" used to calculate the angle of displacement followed by theta = np.degrees(theta), to convert rad to degree. Alpha is employed to line this objects to the axes. The perform angles = XY, scale_units = xy generated to plot the vectors within the x and y plane with its units. I set different colours to properly scan the trajectory of the eagle's flight. plt.legend represents its axis and its label. Plt.show() perform can show all the plotting of the eagle's flight. To more understand the method of the code I created a flow chart diagram. See figure 1 , 1.1, 1.2, 1.3 and 1.4 for the flow chart, code, and result of eagle's flight.

$$dist\ total\ = (longtitude\ total)\hat{x} + (latitude\ total)\hat{y}$$

$$disp = \sqrt{distx^2 + disty^2}$$

$$\theta = \tan^{-1}(\alpha + \frac{y}{x})$$

Formulas for eagle's flight trajectory

[37]:
```
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=3)
    lat = np.random.randint(-10,10, size=3)
    dist1  = np.array([lat[0], long[0]])
    dist2  = np.array([lat[1], long[1]])
    dist3  = np.array([lat[2], long[2]])

    dist_test_1_2 = np.add(dist1,dist2)
    dist_total = np.add(dist_test_1_2,dist3)
    disp = np.sqrt ((dist_total[0]**2) + (dist_total[1]**2))
    alpha = 10**-6
    theta =  np.arctan(dist_total[1]/(alpha+ dist_total[0]))
    theta = np.degrees(theta)

    n=2
    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()



    plt.quiver(0,0, dist1[0], dist1[1],
               angles='xy', scale_units='xy',scale=1, color='red',
               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='blue',
              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='green',
              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='orange',
              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: Eagle's flght trajectory code.

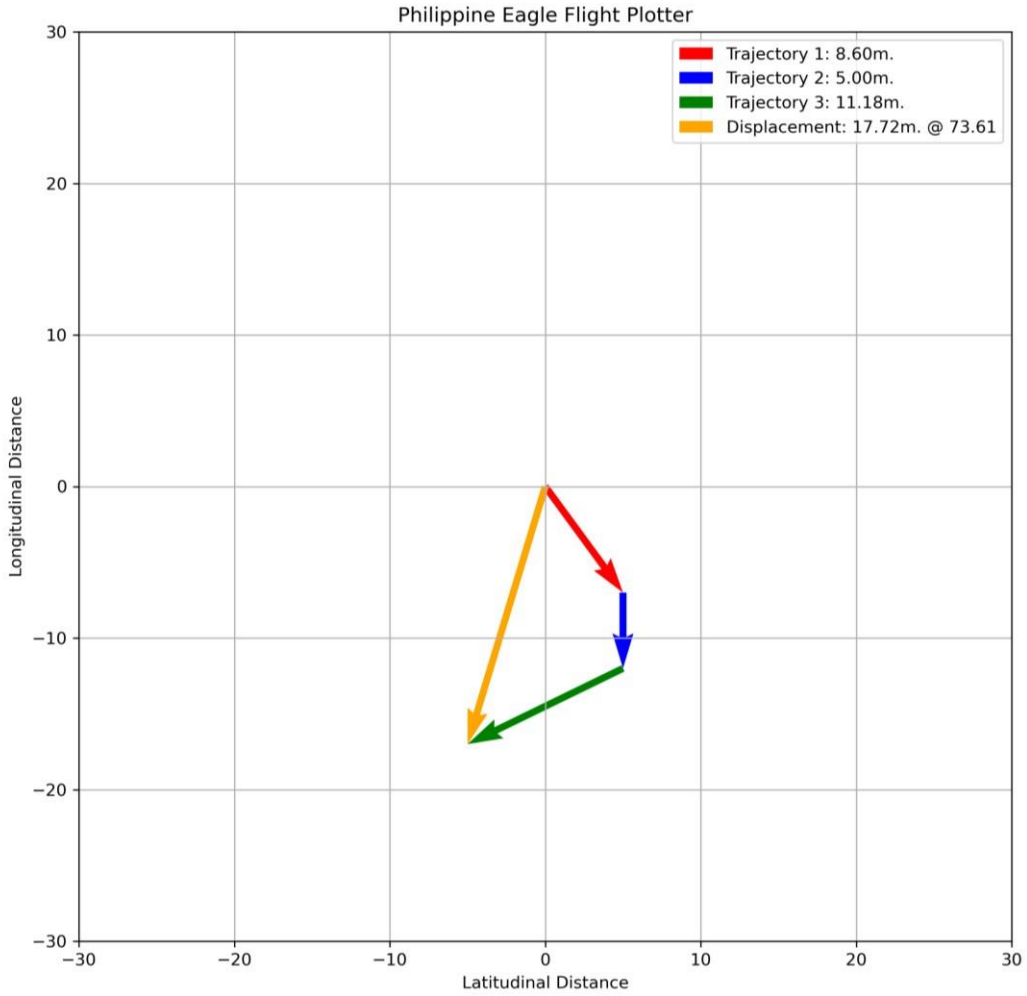

Figure 1.1: First result of eagle's flight trajectory

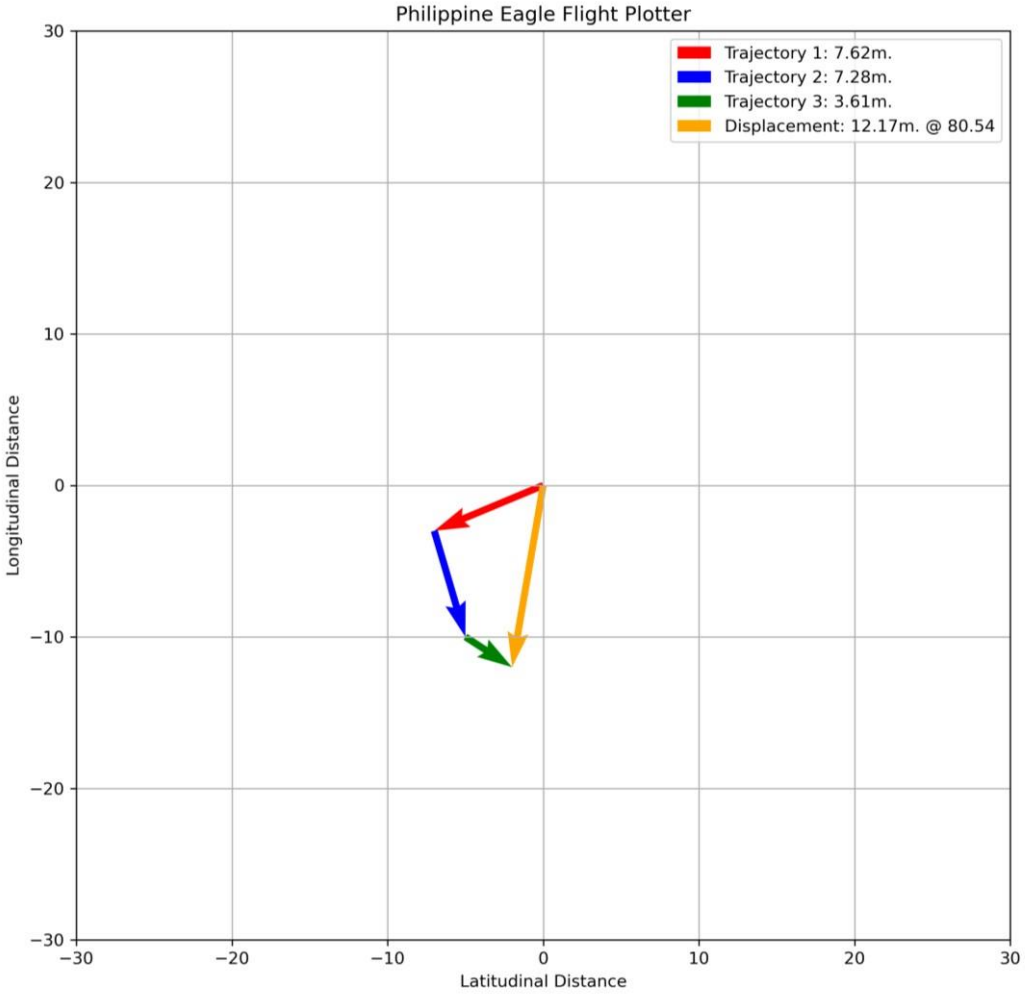

Figure 1.2: Second result of eagle's flight trajectory

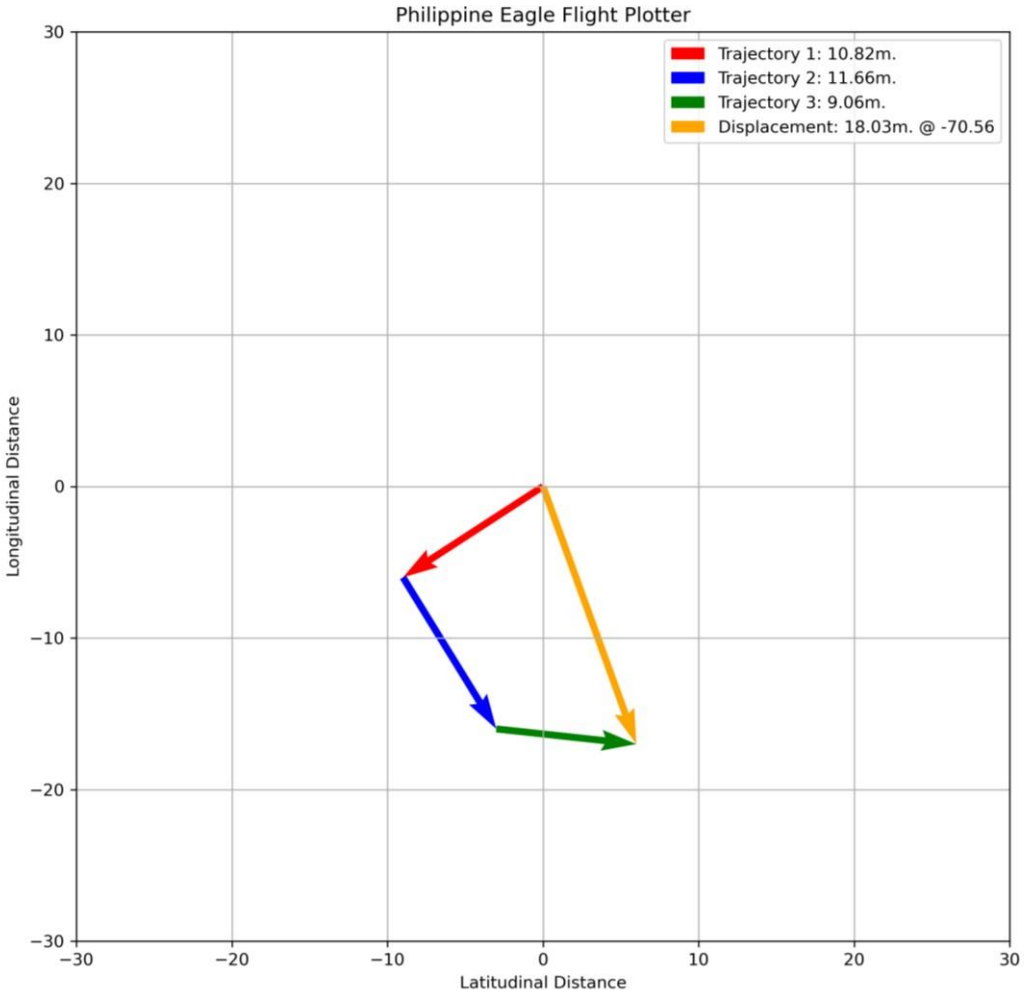

Figure 1.3: Third result of eagle's flight trajectory

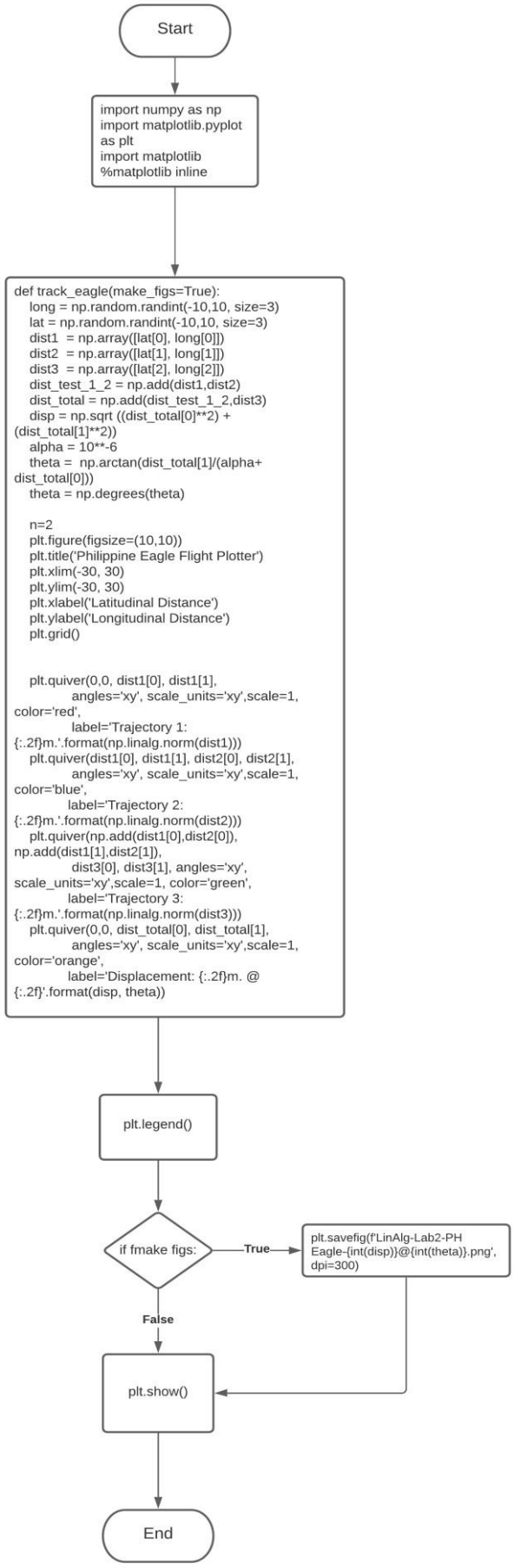

Figure 1.4: Flowchart for eagle's flight trajectory

The second problem is to fix the right documentation of the eagle_kinematics, I utilized in assigning the variables for the time and position of the eagle_kinematics. The functions in eagle_kinematics are to induce the eagle_total position, eagle_total_velocity, and also the eagle_total_acceleration. I assigned pst and res for the input eagle_position and time. Multiply the arrays of ([8,4,2,1] and [2,1,3,2]) is corresponding to the add of [16,4,6,2] is equivalent to 28. the method in obtaining the eagle_total_position of the eagle's flight, followed by its eagle_total_velocity it multiplies the arrays of ([4,2,1] and [6,2,3]) is equivalent to the sum of [24,4,3] that will be such as 31, whereas the eagle_total_acceleration multiply the elements within the arrays ([2,1] and [12,2]) is equivalent to the total of [24,2] that are going to be equivalent to 26. importation numpy as np can use the operate np.array(), wont to produce and store arrays. np.multiply(), used to multiply the corresponding elements. np.sum(), is employed to get all the results of arrays from the axis. For the last function np.zeros() used to come to all or any arrays that store zeroes.

See figure 2 and 2.1

```
import numpy as np
def eagle_kinematics(eagle_position, time):

    req_shape = 4
    eagle_velocity = np.zeros((req_shape-1,))
    eagle_acceleration =  np.zeros((req_shape-2,))
    eagle_total_vector = np.array([res**3, res**2, res, 1])

    if eagle_position.shape == (req_shape,):
        eagle_velocity= np.array([3*eagle_position[0],2*eagle_position[1], eagle_position[2]])
        eagle_acceleration  = np.array([2*eagle_velocity[0],eagle_velocity[1]])
        eagle_total_position = np.sum(np.multiply(eagle_position,eagle_total_vector))
        eagle_total_velocity = np.sum(np.multiply( eagle_velocity, eagle_total_vector[1:]))
        eagle_total_acceleration = np.sum(np.multiply(eagle_acceleration, eagle_total_vector [2:]))

    else:
        print(f'Input displacement vector is not valid. Make sure that the vector shape is ({req_shape},)')

    return eagle_total_position, eagle_total_velocity , eagle_total_acceleration


pst = np.array([2,1,3,2])
res = 2
print("Eagle kinematics:",eagle_kinematics(pst,res))
```

Figure 2: Eagle kinematics code

```
Eagle kinematics: (28, 31, 26)
```

Figure 2.1: Output for the Eagle kinematics

The last problem is obtaining Bebang's online FB to reach post and profit by mistreatment an equivalent idea from the primary problem. however here during this problem it talks regarding the weeks, profit, and reach gradient. I allotted week_1_2 to at week1 and 2. Then another for weeks three and four to induce resultant vectors. I used the mathematician formula to get the week_performance of her online business as function np.sqrt()((week_total[0]**2) + (week_total[1]**2)). Then I set the plt.xlim as (0,1.01*np.sum(reach)) and for the coordinate axis as plt.ylim(np.sum(np.abs(profit)),np.sum(np.abs(profit))). Then for plotting the weeks I used plt.quiver() initiating from its origin (0,0) followed by the week[0] and week[1]. And getting its angles as function angles='xy' and scale units, then I set its arrows color and set the width to 0.0025. Then to get its efficiency, I assigned it first from its origin (0,0), then I inputted the corresponding result as week_total[0], week_total[1]. In order to show the result of the output assigned different values to the profit and FB reach post and to save the image of the result the if makes figs should be true and to print the results, I used the function plt.show(). See figure 3, 3.1, 3.2, and 3.3 for the codes and output.

$$week\_performance = \sqrt{week\_total[0]^2 + week\_total[1]^2}$$

$$reach\_gradient = \tan^{-1}(\alpha + \frac{week\_total[1]}{week\_total[0]})$$

Formula for the week_performance and reach_gradient

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

%matplotlib inline
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_1_2 = np.add(week1,week2)
        week_3_4 =np.add(week3,week4)
        week_total = np.add(week_1_2,week_3_4)
        week_performance =np.sqrt ((week_total[0]**2) + (week_total[1]**2))
        alpha = 10**-6
        reach_gradient = np.arctan((week_total[1]) / (alpha + week_total[0]))
        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()
        n = 2


        plt.quiver(0,0, week1[0], week1[1],
                   angles='xy', scale_units='xy',scale=1, color='blue', 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='crimson', 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='coral', 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='magenta', 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='cyan', 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('Dimension error')


profit= np.array([-1000, 19000, 22000, 900])
reach = np.array([2000, 175, 99, 665])

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

Figure 3: Bebang's Online Business code

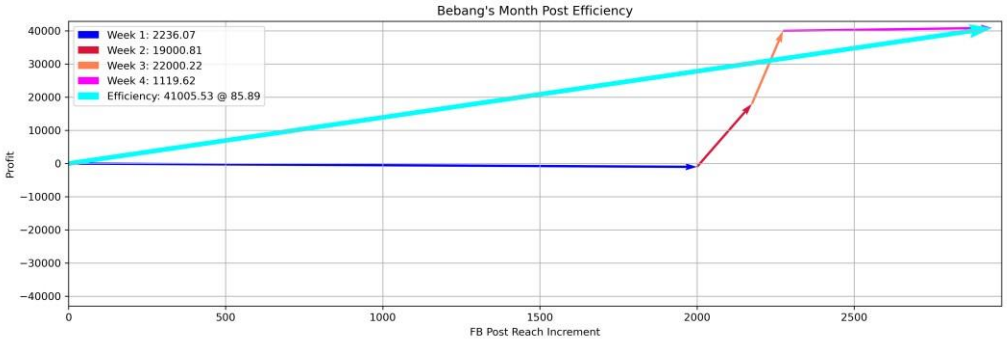

Figure 3.1: First scenario output

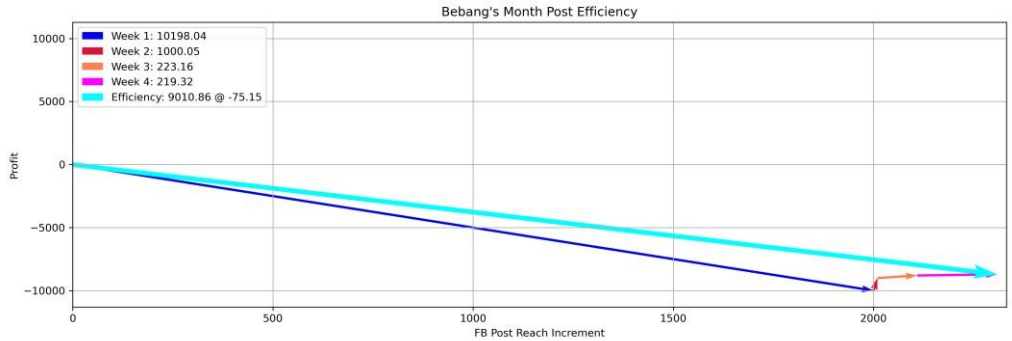

Figure 3.2: Second scenario output

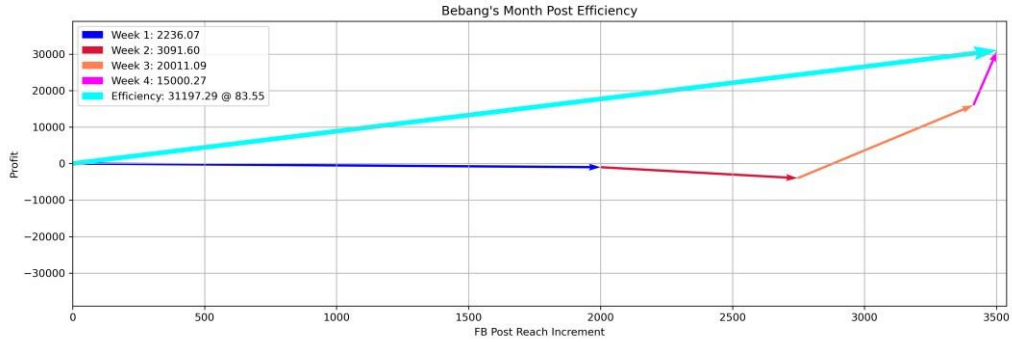

Figure 3.3:Third scenario output

IV. Conclusion

Based on this laboratory no.2, the function I used to imported from numpy and mathplotlib like plt.quiver(), wont to displays its velocity vectors described as arrows with its corresponding element values as (u,v) that may set points of the x and y axis that enforced in obtaining the trajectory of the eagle's flight and in plotting the profit, FB reach posts and potency of Bebang's on-line business.[1]. The numpy as np set large numeric data sorts within the arrays. which is employed in adding the vectors and getting its magnitude for displacement in the initial drawback similar to the third problem and obtaining the values of eagle kinematics.[2]. within the problem of 1 and three, so as to urge its magnitude, I used the Pythagorean formula function as, np.sqrt() [3]. Then I used tan inverse to get the angle of every vector inexplicit to the matter one and three. The alpha variable is employed to line the present axes of objects and to avoid the error in the program [4]. The function plt.figures() wont to set all the current values of arrays.[5]. Plt x and y lim, set the current limit of the x and y-axis in the vectors implied to the problem one and three. [6] Plt.grid() want to set the visibility of the axes from its objects like shown within the results of the flow of arrow directions in the graph with the various colors described in every

displacement and FB reach posts, efficiency, and profits [7]. Plt legend(), used to setting and visualizing the weather of the graph followed by its plt.label() providing its label plot that shown in each end in the matter one and three.[8] Shapes are used to set what percentage rows and columns gift in the vector [9] and therefore the size used to present the total range of elements within the vector [10]. Plt.show(), is employed to show all the potential values in the arrays to get the visual illustration of the graph while not this, I can't ready to scan the results and understand the method of the vectors in the program. Vectors are composed of magnitude and directions, it's gift in our daily lives without appreciating the wonder of it. It applies in our daily lives adore walking from home to a different place, it calculates the direction of your path and over an quantity of your time or a collision of a car and a truck leading to its speed, force, and direction. One of the examples primarily based from one of the web sites I read, is once the boat is crossing a river the current of the water forces the boat in another direction, thus if the boat has its motor, the force are going to be present within the scenario as a result of it's getting into one direction. so as to understand wherever the boat can go, we want to induce the results of the magnitude and direction of every force [12]. Another example given a situation regarding the cup location and table location you have to raise a cup of low together with your eyes closed, the cup is within the table and it's at a one-arm distance far from you. Therefore, it'll take one or two of tries before you lift the cup because you'll end up by swinging your one arm in an arc to find the cup [13].

References

- [1] " Quiver – (MATHLAB Functions)" http://www.ece.northwestern.edu/local-apps/matlabhelp/techdoc/ref/quiver.html (accesed Oct 4,2020)
- [2] "Numpy – (Python Numpy Tutorial with Jupyter and Colab))" https://cs231n.github.io/python-numpy-tutorial/(accessed Oct 4,2020)
- [3] "np.sqrt() – (Geeksforgeeks)" https://www.geeksforgeeks.org/numpy-sqrt-in-python/(accessed Oct 4,2020)
- [4] "Alpha – (Math works)" https://www.mathworks.com/help/matlab/ref/alpha.html (accessed Oct 4,2020)
- [5] " plt.figure() – (What is the necessity of plt.figure() in matplotlib?)" https://stackoverflow.com/questions/38666527/what-is-the-necessity-of-plt-figure-in-matplotlib (accessed Oct 4,2020)
- [6] " pltxandylim() – (How to Set Limits for Axes in Matplotlib | Delft Stack)" https://www.delftstack.com/howto/matplotlib/how-to-set-limit-for-axes-in- (accessed Oct 4,2020) matplotlib/#:~:text=Xlim()%20and%20Ylim()%20to%20Set%20Limits%20of%20Axes%20in%20Matplotlib,matplotlib.&text=xlim()%20and%20matplotlib.,range%20of%20the%20respective%20axes.(acce ssed Oct 4,2020)
- [7] " grid – (Matplotlib-Grids)" https://www.tutorialspoint.com/matplotlib/matplotlib_grids.htm(accessed Oct 4,2020)
- [8] "plt.legends() – (Customizing Plot Legends)" https://jakevdp.github.io/PythonDataScienceHandbook/04.06-customizing-legends.html(accessed Oct 4,2020)
- [9] "Shapes – (numpy.ndarray.shape — NumPy v1.19 Manual)" https://numpy.org/doc/stable/reference/generated/numpy.ndarray.shape.html(accessed Oct 4,2020)
- [10] "Size - (Numpy size() function | Python) " https://www.geeksforgeeks.org/numpy-size-function-python/(accessed Oct 4,2020)
- [11] 'Plt.show() – (Visualization with Matplotlib)" https://jakevdp.github.io/PythonDataScienceHandbook/04.00-introduction-tomatplotlib.html#:~:text=is%20your%20friend.-,plt.,display%20your%20figure%20or%20figures.&text=python%20myplot.py-,The%20plt.,your%20system's%20interactive%20graphical%20backend. (accessed Oct 4,2020)
- [12] " example of vectors in real life situation- (Vectors in space)" https://math.libretexts.org/Bookshelves/Calculus/Book%3A_Calculus_(OpenStax)/12%3A_Vectors_in_Space(accessed Oct 4,2020)
- [13] " another example of vectors in real life situation – (Applications of vectors in real life) " https://www.freeaptitudecamp.com/applications-of-vectors-in-real-life/(accessed Oct 4,2020)

Github link:

https://github.com/LorenzoMiguelColumba/LorenzoMiguelColumba-LinALg_Lab2-.git