Adamson University
College of Engineering
Computer Engineering Department

Linear Algebra

Laboratory Activity No. x

Multidimensional Vectors

Submitted by:
Sustento, Myke Alvin E.

Instructor:
Engr. Dylan Josh D. Lopez

November 06, 2020

I. Objectives

This laboratory activity aims to implement the principles and techniques of representing linear combinations in the 3-dimensional plane and visualizing vectors using the programming language Python.

II. Methods

The practices of the activity consist of using the functions included in the NumPy library and matplotlib library. The functions that were used for task 1 was np.array() to create an array [1] to represent the vectors, np.arange() to return evenly spaced values within a given interval [2] and get the scalar value, plt.plot() to plot y versus x as lines and/or markers [3], and plt.show() to display all the figures [4]. The equation in task 1 was a system of linear equations $A = \begin{cases} 1x + 9y \\ 8x + 11y \end{cases}$. These values would be used by the plt.plot() multiplied with the scalar values.

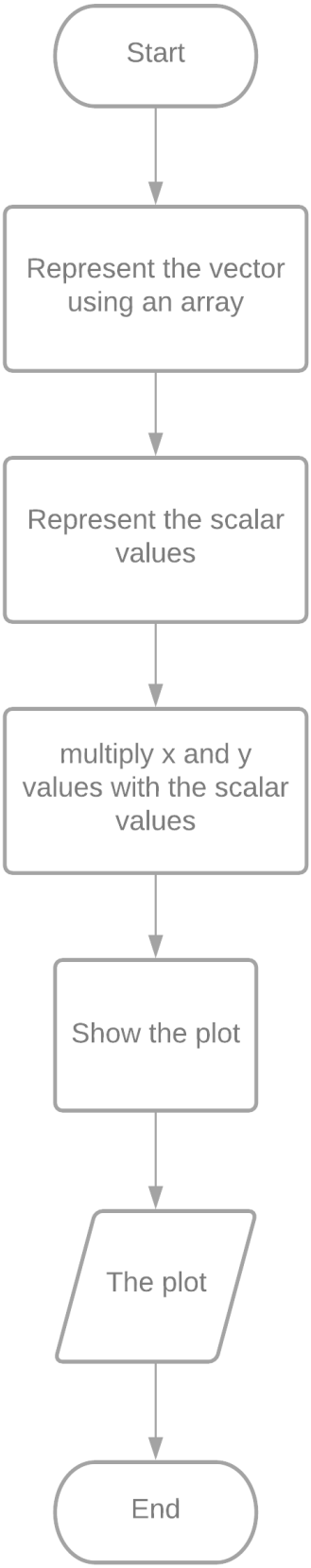

Figure 1 The Flowchart for Task 1

Figure 1 shows the flowchart that would be used in the program for task 1. The program would start by representing the vector with an array using np.array() and representing the scalar values using np.arange(). Afterward, all the x and y values from the system of linear equation will be multiplied with the scalar values and plotted using a looped plt.plot().

The functions that were used in task 2 were np.array() to create an array[1] that would represent the vectors, np.arange() to return evenly spaced values within a given interval [2] and get the scalar value, plt.figure()to create a new 3d figure [5], plt.gca to get the current axes with the projection = '3d' [6], plt.plot() to plot y versus x as lines and/or markers [3], and plt.show() to display all the figures [4]. The equation in task 2 was a system of linear equations $A = \begin{cases} 8x + 5y + 6z \\ 2x + 1y + 3z \\ 4x + 5y + 2z \end{cases}$. These values would be used by the plt.plot() multiplied with the scalar values.

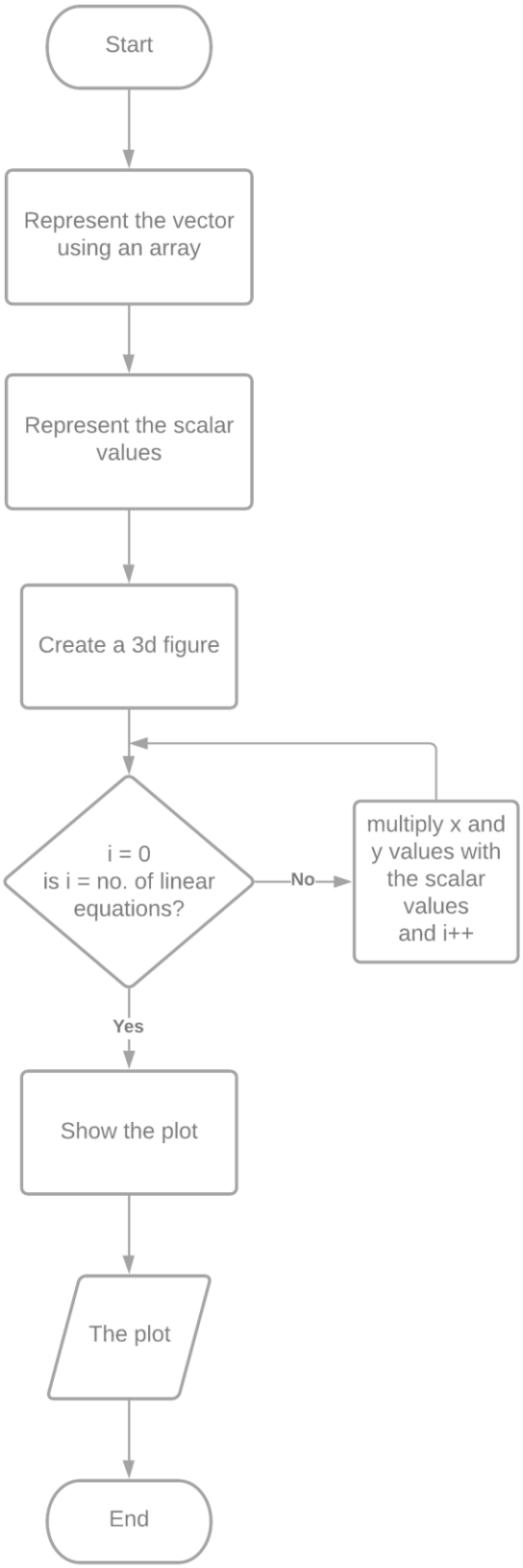

Figure 2 The Flowchart for Task 2

Figure 2 shows the flowchart that would be used in the program for task 2. The program would start by representing the vector with an array using np.array() and representing the scalar

values using np.arange(). Since the vectors are in a 3d plane, a 3d figure will be created. Afterward, all the x and y values from the system of linear equation will be multiplied with the scalar values and plotted using a looped plt.plot().

The practices were used in order to achieve the deliverables of the activity which is to represent linear combinations in the 3-dimensional plane and visualize vectors using the programming language Python.

III. Results

```
### TYPE YOU CODE FOR TASK 1 HERE
# 2d vectors
A = np.array([
    [1, 9],
    [8, 11]
    ])

# scalar value
c = np.arange(0,100)

# plot for 1st vector
plt.plot(c*A[0,0], c*A[0,1], color='red')

#plot for 2nd vector
plt.plot(c*A[1,0], c*A[1,1], color='blue')

#show figures
plt.show()
```

Figure 3 The Program for Task 1

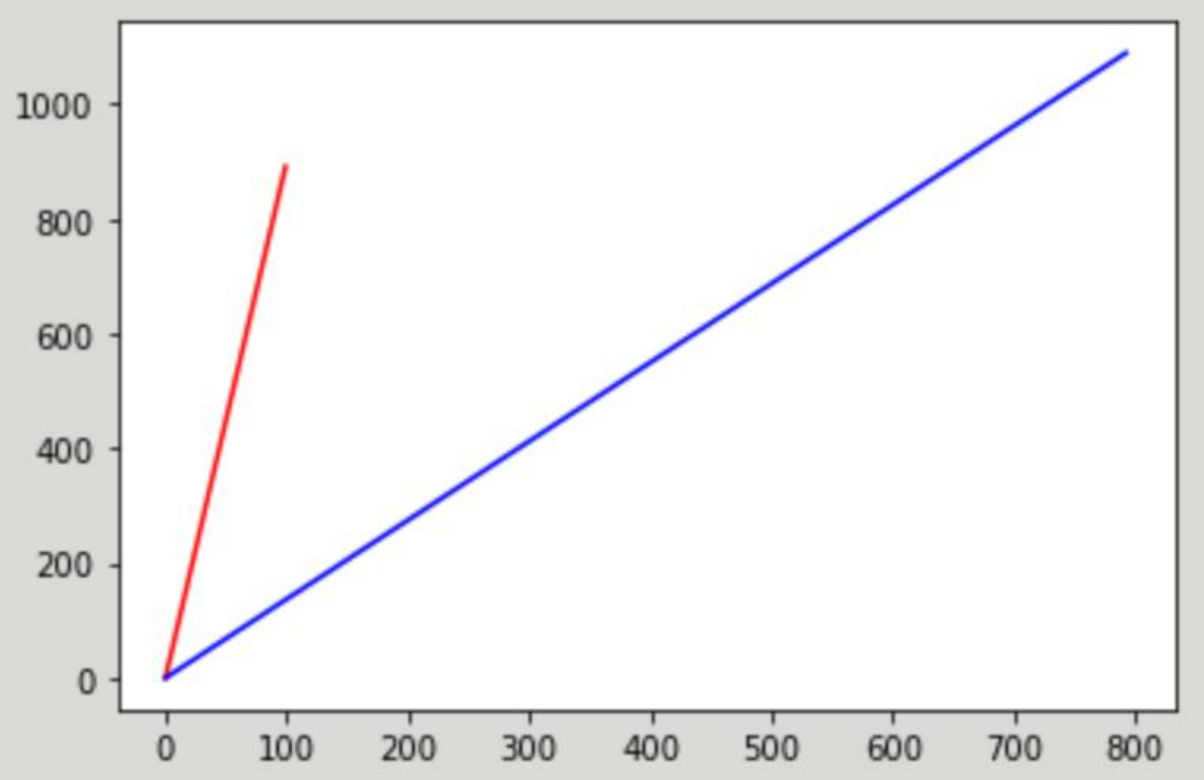

Figure 4 The Plot Result of Task 1

Figure 4 shows the plot created by the program in figure 3. Since the vector was a $\mathbb{R} = 1$, the dimension of the plot was one-dimensional which would look like a straight line. The equation was $A = \begin{cases} 1x + 9y \\ 8x + 11y \end{cases}$ therefore the array that was created using np.array() was $[1, 9]$. The scalar value that was created using np.arange() were (0,100). After multiplying the scalar value with the array and creating a plot using plt.plot(), plt.show() was used to display the figures. The displayed figure would show 2 plotted lines since there were 2 vectors. The 1st line which is represented as a red line went from (0, 0) to (100, 900) while the 2nd line which is represented as a blue line went from (0,0) to (800, 1100). The endpoint values of these lines were the product of the scalar value and the array.

```
### TYPE YOU CODE FOR TASK 2 HERE
# 3d vectors
A = np.array([
    [8,5,6],
    [2,1,3],
    [4,5,2]
])

# scalar values
c = np.arange(0,3)

# create a 3d figure
fig = plt.figure()
ax1 = fig.gca(projection='3d')

# plot for the 3 vectors
colors = ['g','r','b']
for i in range(A.shape[0]):
    #for a number rows, plot it to the corresponding column values
    ax1.plot(c*A[i,0],c*A[i,1],c*A[i,2], color=colors[i])

# show figure
plt.show()
```

Figure 5 The Program for Task 2

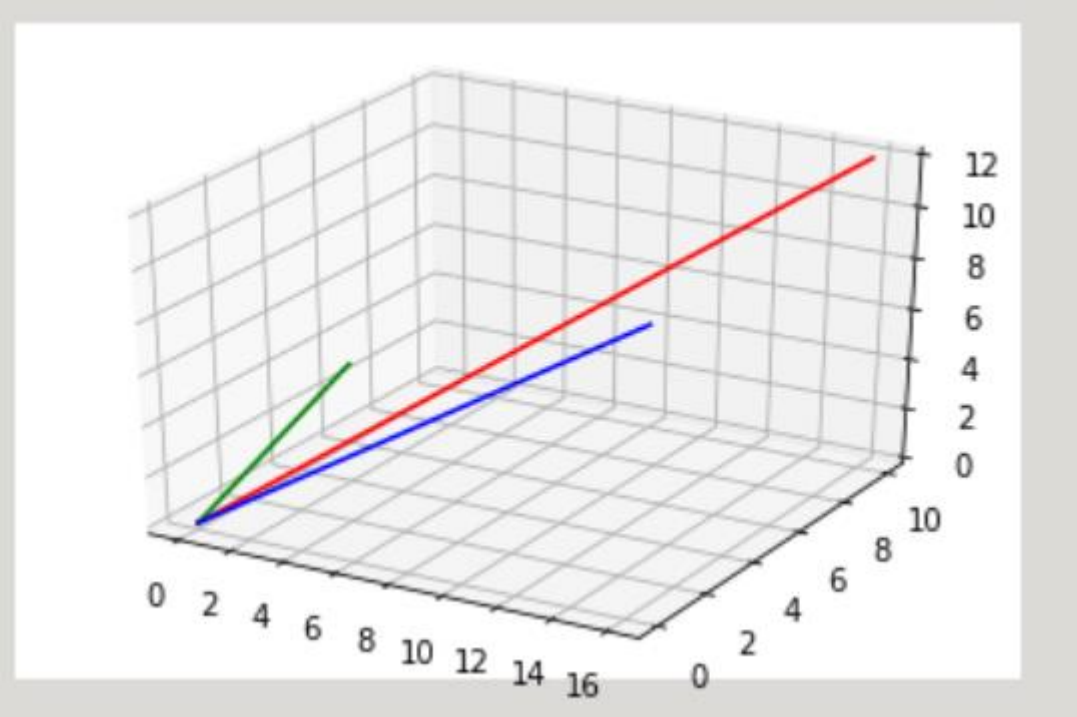

Figure 6 The Plot Result of Task 1

Figure 6 shows the plot created by the program in figure 5. Since the $\mathbb{R} = 2$, The result would look like a 2-dimensional line but since the projection that was used was 3d, the result

was plotted in a 3d plane. $A = \begin{cases} 8x + 5y + 6z \\ 2x + 1y + 3z \\ 4x + 5y + 2z \end{cases}$ therefore the array that was created using np.arrray() was $\begin{bmatrix} 8 & 5 & 7 \\ 2 & 1 & 3 \\ 4 & 5 & 2 \end{bmatrix}$. The scalar value that was created using np.arange() were (0,3). In order to create a 3d figure, plt.figure() and fig.gca(projection = '3d') was used. Afterward, a loop was created to show the 3 vector plot by multiplying the scalar values and the vectors in the array. Lastly, plt.show() was used to display the figure. The result would be 3 2d lines in a 3-dimensional plane. The endpoint values of these lines were the product of the scalar value and the array.

Other types of data that can be plotted in the 2-d or 3-d plane are ordinal data. Ordinal values are discrete and ordered units so it can be summarized using frequencies, proportions, percentages, and visualized using pie charts and bar charts [7].

Data can have more than 3 dimensions but since 4 dimensions can't be experienced by the limitation of a human's physical sense, it is represented in a 3-dimensional plane such as the examples in [8]

IV. Conclusion

This laboratory proves that linear combinations and vectors can be represented and visualized in a 3-dimensional plane. Vectors can be represented by systems of linear equations and converted into a plot with the help of the functions in the NumPy and matplotlib libraries. Knowledge about dimensions above 3 was also included. To conclude the laboratory report, vectors can be represented multidimensionally.

The concept of visualizing and representing vectors in n-dimensions for business applications can be used in products that are to be sold. The "The Art of Effective Visualization of Multi-dimensional Data" showed an analysis of food products specifically wine [8]. Businesses can apply this to improve their products and sell to more consumers.

References

[1]"numpy.array — NumPy v1.19 Manual", Numpy.org, 2020. [Online]. Available: https://numpy.org/doc/stable/reference/generated/numpy.array.html. [Accessed: 01- Nov- 2020].

[2]"numpy.arange — NumPy v1.19 Manual", Numpy.org, 2020. [Online]. Available: https://numpy.org/doc/stable/reference/generated/numpy.arange.html. [Accessed: 25- Oct- 2020].

[3]"matplotlib.pyplot.plot — Matplotlib 3.3.2 documentation", Matplotlib.org, 2020. [Online]. Available: https://matplotlib.org/3.3.2/api/_as_gen/matplotlib.pyplot.plot.html. [Accessed: 06- Nov- 2020].

[4]"matplotlib.pyplot.show — Matplotlib 3.3.2 documentation", Matplotlib.org, 2020. [Online]. Available: https://matplotlib.org/3.3.2/api/_as_gen/matplotlib.pyplot.show.html. [Accessed: 25- Oct- 2020].

[5]"matplotlib.pyplot.figure — Matplotlib 3.3.2 documentation", Matplotlib.org, 2020. [Online]. Available: https://matplotlib.org/3.3.2/api/_as_gen/matplotlib.pyplot.figure.html. [Accessed: 06- Nov- 2020].

[6]"matplotlib.pyplot.gca — Matplotlib 3.1.2 documentation", Matplotlib.org, 2020. [Online]. Available: https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.pyplot.gca.html. [Accessed: 06- Nov- 2020].

[7]"Data Types in Statistics", Medium, 2020. [Online]. Available: https://towardsdatascience.com/data-types-in-statistics-347e152e8bee. [Accessed: 06- Nov- 2020].

[8] D. Sarkar, "The Art of Effective Visualization of Multi-dimensional Data", Medium, 2020. [Online]. Available: https://towardsdatascience.com/the-art-of-effective-visualization-of-multi-dimensional-data-6c7202990c57#:~:text=Visualizing%20data%20in%20Four%20Dimensions%20(4%2DD)&text=One%20way%20to%20visualize%20data,plot%20like%20a%20scatter%20plot.&text=Another%20strategy%20is%20to%20keep,point%20size%20as%20data%20dimensions. [Accessed: 06- Nov- 2020].

Appendix

Github Link - https://github.com/Sus102/LA_Lab/tree/master/LA_LAB_5