Graficacion_Vectores_1

September 29, 2024

1 Universidad Autónoma del Estado de México

2 Centro Universitario UAEM Zumpango

2.1 Ingenierá En Computación

2.2 Graficacion Computacional.

Alumno: Jesus Enrique Lugo Ramirez

Profesor: Hazem alvarez

Fecha: 10 de Septiembre del 2024

2.3 Descripcion: GRAFICACION DE VECTORES.

3 INTRODUCCION

3.1 Un vector es una secuencia ordenada de números, que puede entenderse como una lista de valores en una dimensión. En matemáticas, los vectores se utilizan para representar magnitudes y direcciones en el espacio, y son esenciales en áreas como la física, geometría y computación. En programación, los vectores pueden ser usados para representar datos de manera estructurada y realizar cálculos matemáticos como suma, multiplicación, etc.

3.2 En Python, los vectores se pueden manejar fácilmente usando listas, aunque para tareas más complejas y eficientes se suele emplear la biblioteca NumPy, que permite realizar operaciones vectoriales de manera mucho más rápida y optimizada.

4 DESARROLLO

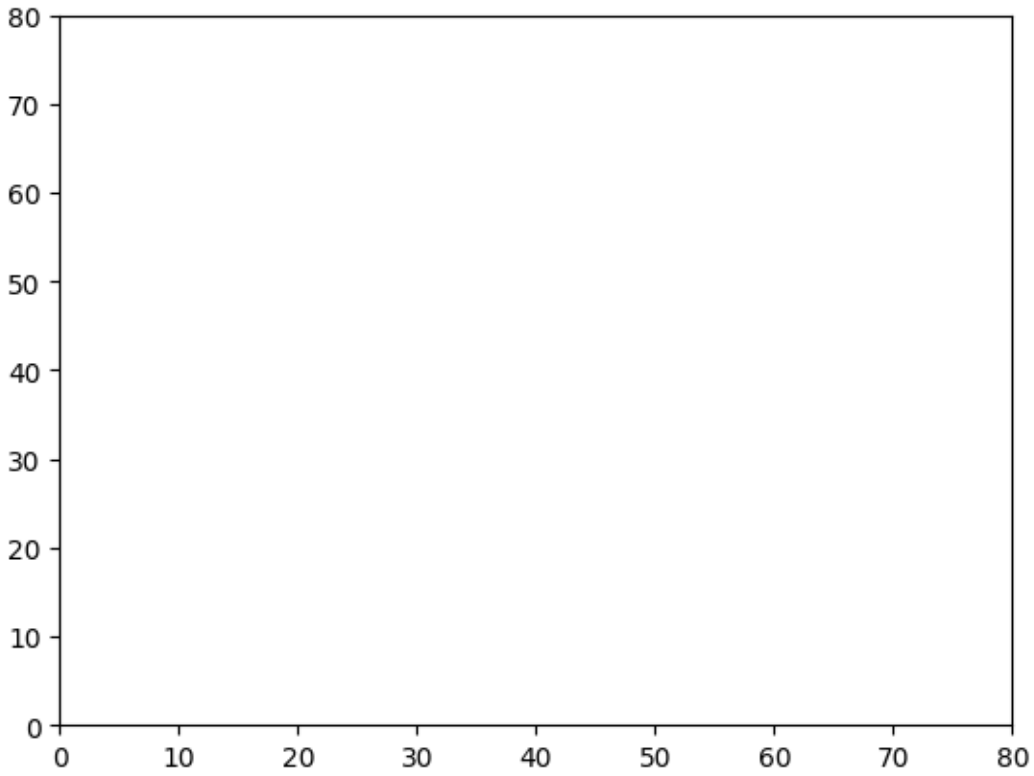
4.1 Utilizando el lenguaje de programacion python se representaran los vectores realizados en clase implementando librerias como numpy o matplotlib.

[11]:
```
#Importando librerias.
import numpy as np
import matplotlib.pyplot as plt
```

5 Creando un plano 2D

[12]:
```
x1 = 0  # Definimos el valor mínimo del eje X
x2 = 80  # Definimos el valor máximo del eje X

y1 = 0  # Definimos el valor mínimo del eje Y
y2 = 80  # Definimos el valor máximo del eje Y

# Editamos los ejes utilizando los valores definidos para X e Y
plt.axis([x1, x2, y1, y2])
```

[12]:
```
(0.0, 80.0, 0.0, 80.0)
```

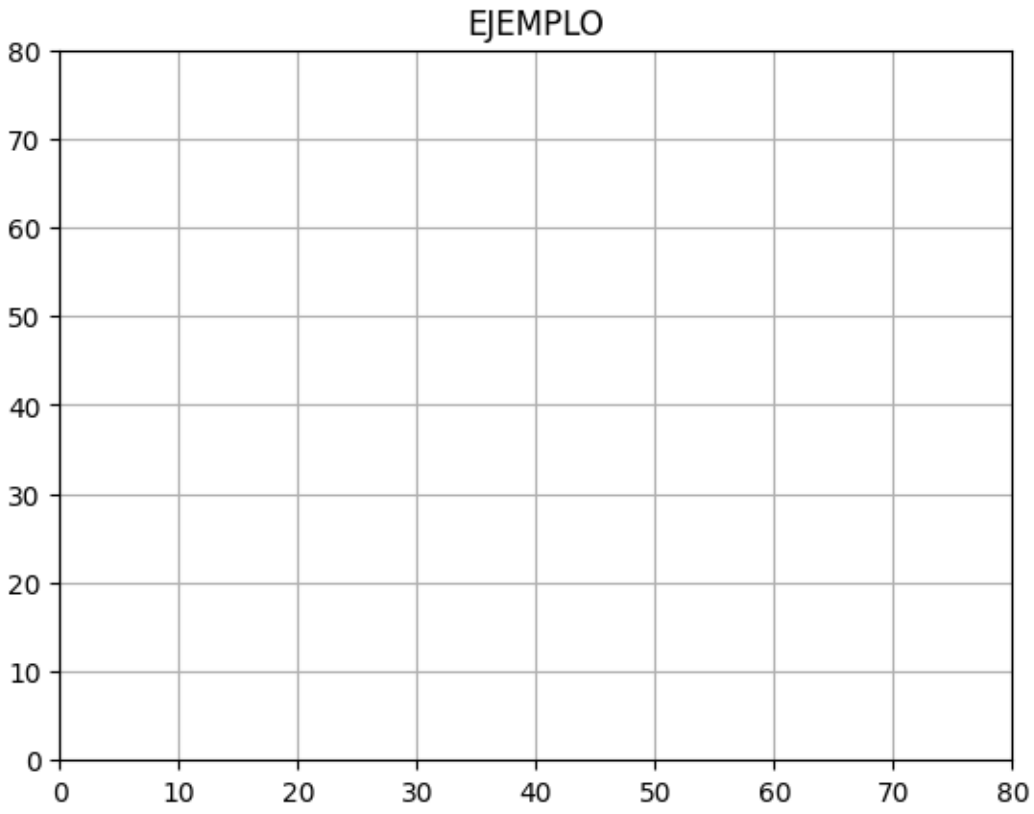


6 Agregando función para visualizar la cuadricula.

[13]:
```
x1 = 0  # Valor inicial (mínimo) para el eje X
x2 = 80  # Valor final (máximo) para el eje X

y1 = 0  # Valor inicial (mínimo) para el eje Y
y2 = 80  # Valor final (máximo) para el eje Y
```

```
# Editamos los ejes utilizando los valores definidos para X e Y
plt.axis([x1, x2, y1, y2])

# Activamos la visualización de los ejes
plt.axis('on')

# Mostramos la cuadrícula en el gráfico
plt.grid(True)

# Agregamos un título al gráfico
plt.title("EJEMPLO")
```

[13]: Text(0.5, 1.0, 'EJEMPLO')

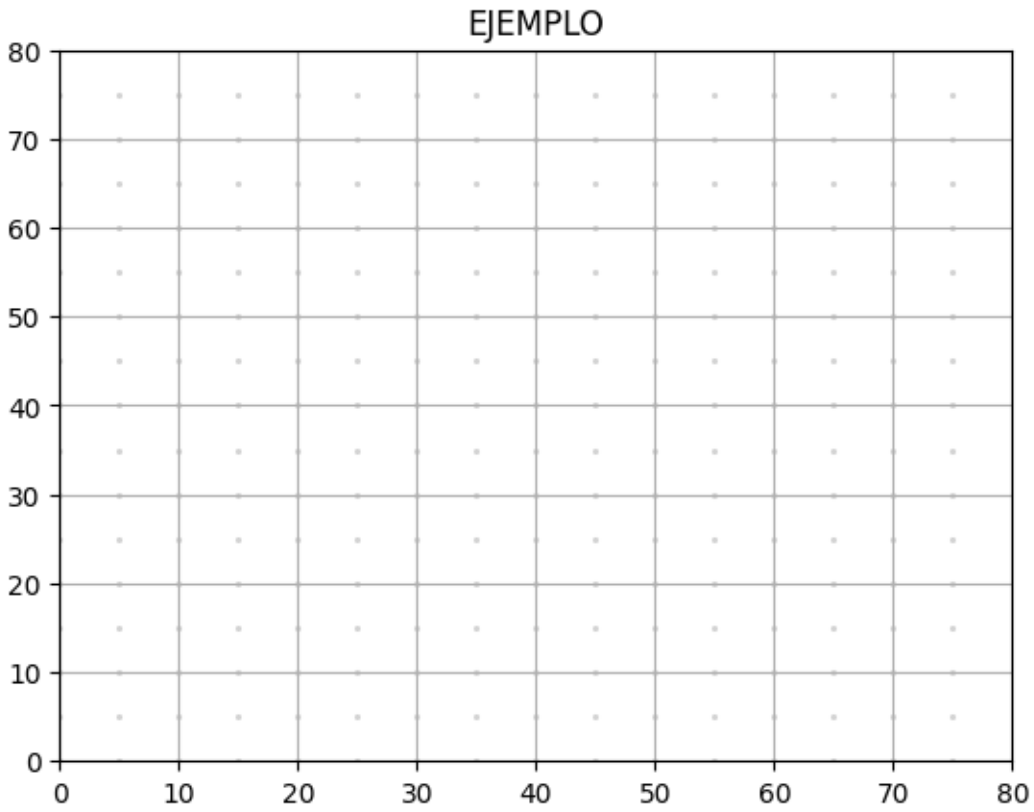


7 Implementando funcion para graficar puntos en el plano.

[14]:
```
# Imprimimos un mensaje en la consola
print("espacio xd")

x1 = 0  # Valor mínimo para el eje X
```

```
x2 = 80  # Valor máximo para el eje X

y1 = 0  # Valor mínimo para el eje Y
y2 = 80  # Valor máximo para el eje Y

# Editamos los ejes del gráfico
plt.axis([x1, x2, y1, y2])

# Activamos la visualización de los ejes
plt.axis('on')

# Activamos la cuadrícula en el gráfico
plt.grid(True)

# Asignamos un título al gráfico
plt.title("EJEMPLO")

# Definimos la distancia entre puntos en el eje Y
dy = 5

# Definimos la distancia entre puntos en el eje X
dx = 5

# Graficar puntos en el plano
for x in np.arange(x1, x2, dx):  # Iteramos sobre el rango de valores para X
    for y in np.arange(y1, y2, dy):  # Iteramos sobre el rango de valores para Y
        # Graficamos los puntos con coordenadas (x, y)
        plt.scatter(x, y, s=1.5, color='lightgray')  # s es el tamaño del punto␣
 ↪y color es el color
```

espacio xd



8 Creando el primer vector de ejemplo

[15]:
```
# Graficando el vector

x1 = 0  # Valor mínimo para el eje X
x2 = 80  # Valor máximo para el eje X

y1 = 0  # Valor mínimo para el eje Y
y2 = 80  # Valor máximo para el eje Y

# Editamos los ejes del gráfico
plt.axis([x1, x2, y1, y2])

# Activamos la visualización de los ejes
plt.axis('on')

# Activamos la cuadrícula en el gráfico
plt.grid(True)

# Asignamos un título al gráfico
```

```
plt.title("EJEMPLO")

# Definimos la distancia entre puntos en el eje Y
dy = 5

# Definimos la distancia entre puntos en el eje X
dx = 5

# Graficar puntos en el plano
for x in np.arange(x1, x2, dx):  # Iteramos sobre el rango de valores para X
    for y in np.arange(y1, y2, dy):  # Iteramos sobre el rango de valores para Y
        # Graficamos los puntos con coordenadas (x, y)
        plt.scatter(x, y, s=1.5, color='lightgray')  # s es el tamaño del punto, color es el color del punto

# Graficar una flecha (vector)
# plt.arrow(inicio_x, inicio_y, desplazamiento_x, desplazamiento_y, head_length=tamaño de la punta, head_width=ancho de la punta, color=color)
plt.arrow(0, 75, 10, 0, head_length=4, head_width=4, color="k")  # Flecha que inicia en (0,75), de longitud 10 en el eje X, con una punta de longitud 4 y ancho 4, de color negro ("k")
```

[15]: <matplotlib.patches.FancyArrow at 0x27dbcd04c80>