**1. Importación de bibliotecas**

import numpy as np

import matplotlib.pyplot as plt

from scipy.interpolate import BSpline, make\_interp\_spline

* numpy: Para manejar arreglos y cálculos numéricos.
* matplotlib.pyplot: Para graficar la curva y los puntos de control.
* scipy.interpolate.BSpline: Para generar la curva B-spline.
* make\_interp\_spline: No se usa en este código, pero sirve para generar interpolaciones suaves (puede eliminarse si no se usa).

**2. Definir los puntos de control**

control\_points = np.array([(0, 0), (1, 2), (3, 3), (4, 0), (5, 2)])

n = len(control\_points) - 1 # Número de segmentos entre los puntos

* control\_points: Define los puntos que guían la forma de la curva.
* n: Calcula cuántos segmentos hay entre los puntos de control.

**3. Crear el vector de nudos**

k = 3 # Grado de la B-spline (cúbica)

knot\_vector = np.concatenate(([0] \* k, np.linspace(0, 1, n - k + 2), [1] \* k))

* k = 3: Define una **B-spline cúbica** (grado 3).
* knot\_vector: Define el conjunto de **nudos** (valores donde se definen los segmentos de la curva).
  + Se repiten k ceros al inicio y k unos al final para suavizar los extremos.
  + np.linspace(0, 1, n - k + 2): Genera los valores internos entre 0 y 1.

**4. Separar coordenadas X e Y**

x, y = control\_points[:,0], control\_points[:, 1]

* Extrae las coordenadas **X** e **Y** de los puntos de control.

**5. Crear las funciones de la B-spline**

spline\_x = BSpline(knot\_vector, x, k)

spline\_y = BSpline(knot\_vector, y, k)

* BSpline(knot\_vector, x, k): Crea la función de interpolación para **X**.
* BSpline(knot\_vector, y, k): Crea la función de interpolación para **Y**.

**6. Evaluar la curva en un rango de valores**

python

CopyEdit

t = np.linspace(0, 1, 100)

curve\_x, curve\_y = spline\_x(t), spline\_y(t)

* t = np.linspace(0, 1, 100): Genera 100 puntos equidistantes en el dominio [0,1] para evaluar la curva.
* spline\_x(t), spline\_y(t): Evalúa la B-spline en t para obtener coordenadas curve\_x y curve\_y.

**7. Graficar la curva y los puntos de control**

plt.plot(curve\_x, curve\_y, label='Curva B-spline') # Dibuja la curva B-spline

plt.scatter(x, y, color='red', label='Puntos de control') # Muestra los puntos de control

plt.plot(x, y, linestyle='dashed', color='gray', alpha=0.5) # Une los puntos de control con una línea discontinua

plt.legend()

plt.xlabel('X')

plt.ylabel('Y')

plt.title('Curva B-spline en Python')

plt.show()

* plt.plot(curve\_x, curve\_y): Grafica la curva interpolada.
* plt.scatter(x, y, color='red'): Muestra los puntos de control en rojo.
* plt.plot(x, y, linestyle='dashed'): Conecta los puntos de control con una línea discontinua.
* plt.legend(): Agrega una leyenda.
* plt.xlabel() y plt.ylabel(): Etiquetas de los ejes.
* plt.title(): Título del gráfico.
* plt.show(): Muestra el gráfico.

**Resumen**

Este código:

1. Define **puntos de control**.
2. Crea una **B-spline cúbica** con un **vector de nudos adecuado**.
3. Evalúa la curva en **100 puntos**.
4. **Grafica** la curva con los puntos de control.

