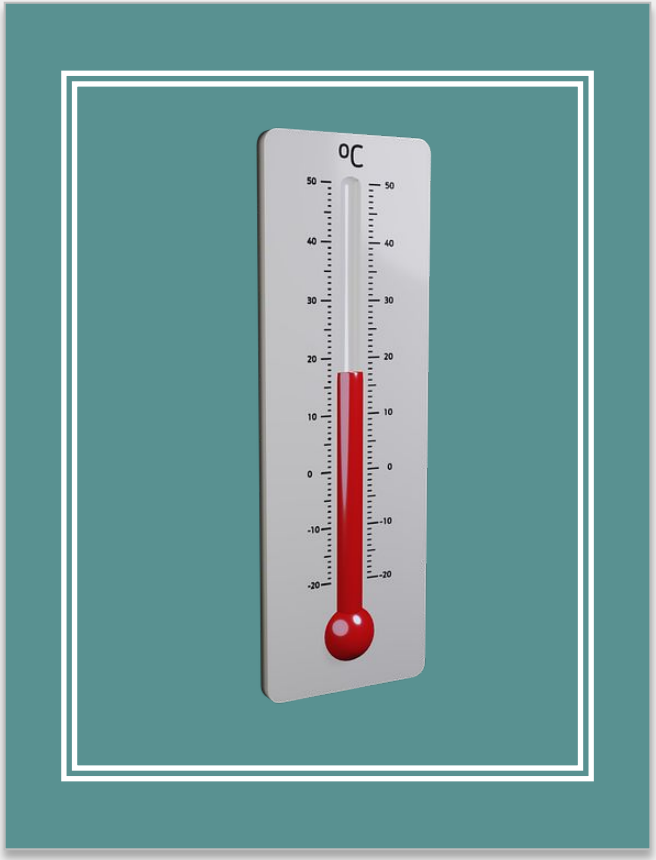
Reto 1- interpolación

Loui Gerard Vélez
Daniel Castellanos
Andrés José Rodríguez Ortega

Descripción del problema

Determinar numéricamente, los valores de la variable climática que están indexados al tiempo y espacio.

Utilizar la interpolación o ajuste de curvas, con el objetivo de poder aproximar la temperatura en puntos determinados.



Metodología utilizada

- Interpolación de Hermite y por spline cúbica utilizando métodos de la librería scipy.
- Utilización de métodos de la librería numpy para el manejo de arreglos y para hallar los errores.
- Métodos de la librería matplotlib para la generación de las gráficas.

$$p(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{1}{2}f''(x_0)(x-x_0)^2 + \frac{f(x_1) - f(x_0) - f'(x_0)(x_1-x_0) - \frac{1}{2}f''(x_0)(x_1-x_0)^2}{(x_1-x_0)^3}(x-x_0)^3$$
$$+ \frac{3f(x_0) - 3f(x_1) + (2f'(x_0) + f'(x_1))(x_1-x_0) + \frac{1}{2}f''(x_0)(x_1-x_0)^2}{(x_1-x_0)^4}(x-x_0)^3(x-x_1)$$
$$+ \frac{6f(x_1) - 6f(x_0) - 3(f'(x_0) + f'(x_1))(x_1-x_0) + \frac{1}{2}(f''(x_1) - f''(x_0))(x_1-x_0)^2}{(x_1-x_0)^5}(x-x_0)^3(x-x_1)^2.$$

$$f(x) = \begin{cases} a_1 x^3 + b_1 x^2 + c_1 x + d_1 & \text{if } x \in [x_1, x_2] \\ a_2 x^3 + b_2 x^2 + c_2 x + d_2 & \text{if } x \in (x_2, x_3] \\ \dots \\ a_n x^3 + b_n x^2 + c_n x + d_n & \text{if } x \in (x_n, x_{n+1}] \end{cases}.$$

Implementación

```
archivo = open("Itatira.csv")
archivo2 = open("Quixada.csv")
i = 0
for linea in archivo:
    s = linea.split(";")
    aux.append(float(s[2]))

fx = np.array(aux)

for linea in archivo2:
    s = linea.split(";")
    aux2.append(float(s[2]))

fx2 = np.array(aux2)
```

```
# Interpolación Hermite
px = PchipInterpolator(x, fx)

muestras = 1000
a = np.min(x)
b = np.max(x)
p_x = np.linspace(a, b+1, muestras)
pfx = px(p_x)
for i in range (len(p_x)):
    p_x[i] = round(p_x[i], 2)
pfx = px(p_x)
for i in range (len(pfx)):
    pfx[i] = round(pfx[i], 2)

# Interpolacion spline cúbica
px = CubicSpline(x, fx)

muestras = 1000
a = np.min(x)
b = np.max(x)
p_x = np.linspace(a, b+1, muestras)
pfx = px(p_x)
for i in range (len(p_x)):
    p_x[i] = round(p_x[i], 2)
pfx = px(p_x)
for i in range (len(pfx)):
    pfx[i] = round(pfx[i], 2)

# Parte 2
px = PchipInterpolator(x2, fx2)
muestras = 1000
a = np.min(x2)
b = np.max(x2)
p_x = np.linspace(a, b+1, muestras)
for i in range (len(p_x)):
    p_x[i] = round(p_x[i], 2)
pfx = px(p_x)
for i in range (len(pfx)):
    pfx[i] = round(pfx[i], 2)
```

Errores

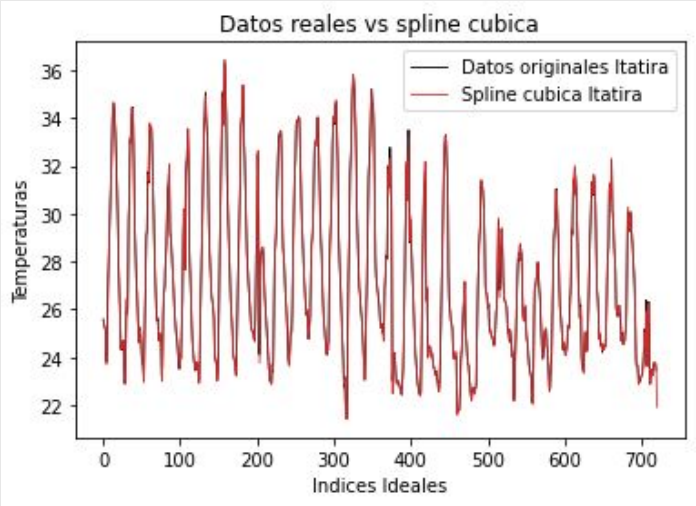
Parte 1	Interpolación Hermite	Spline cúbica
Error máximo	0.5865 (Indice 219)	0.5856 (Indice 219)
Error mínimo	0.0 (Indice 0)	0.0 (Indice 0)
Error medio	0.1474	0.1477
Índice de Jaccard	0.3473	0.3507

Parte 2	Interpolación lineal
Error máximo	0.6201 (Indice 386)
Error mínimo	0.0007 (Indice 683)
Error medio	0.1468
Índice de Jaccard	0.3052

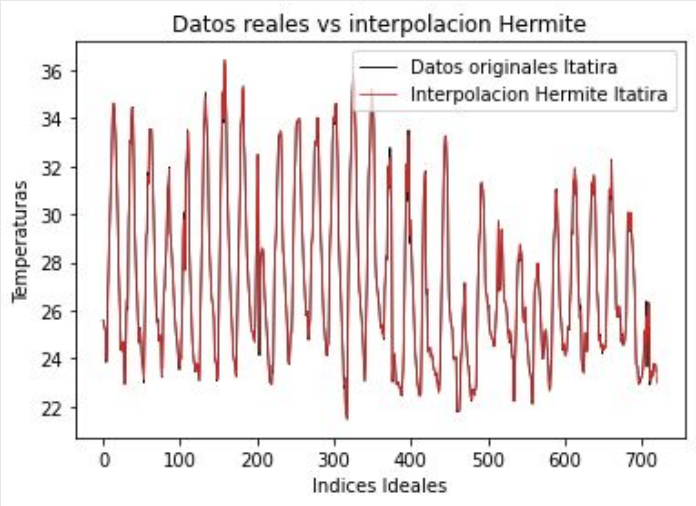
Resultados

Gráfica interpolación:

Spline cúbica



Interpolacion Hermite



Resultados

Itatira vs Interpolados Quixadá