Métodos Numéricos y Optimización: Trabajo Práctico I

Abril Gerbazoni y Lucas Romano Abdo

29 de Marzo de 2024

I Resumen

Resolver problemas de interpolación puede resultar, en ocasiones, tan caro computacionalmente como necesario. Es por esto que, intentamos aprovechar al máximo la información discreta obtenida a través de mediciones para reducir el error a la vez que controlamos la complejidad del problema. Tanto el método de Lagrange como Splines Cúbicos son efectivos en general, pero, durante el transcurso de la investigación buscamos optimizar al máximo sus resultados seleccionando los puntos de nuestro dataset. Utilizando las librerías scipy, numpy y matplotlib de python logramos estudiar el desempeño de las distintas formas de interpolación.

II Introducción

En la vida cotidiana y en diversas aplicaciones, comprender completamente el comportamiento de una función puede resultar desafiante y, a menudo, costoso. En estas situaciones, recurrir a métodos de aproximación se convierte en una herramienta fundamental. Estos métodos buscan modelar la forma de una función a partir de puntos conocidos por los que la función pasa, en un proceso conocido como interpolación.

El objetivo principal de este informe es presentar una comparativa entre varios métodos de interpolación, evaluando su precisión y su capacidad para representar fielmente la función original. Para ello, nos basaremos en diferentes criterios de comparación, que incluyen la cantidad de puntos seleccionados, la disposición equiespaciada o no de dichos puntos, entre otros aspectos relevantes.

III Métodos preexistentes

- Interpolación con el polinomio de Lagrange:

$$P(x) = L_0(x)f(x_0) + L_1(x)f(x_1) \quad (1)$$

$$P(x) = \frac{x - x_1}{x_0 - x_1}f(x_0) + \frac{x - x_0}{x_1 - x_0}f(x_1) \quad (2)$$

- Interpolacion con splines lineales:

  El método de splines lineales implica la aproximación de una función utilizando un conjunto de puntos conocidos. Se traza una línea recta entre cada par de puntos adyacentes, generando así segmentos lineales consecutivos, llamados splines lineales. Este proceso se repite n-1 veces, donde n es la cantidad de puntos conocidos. Cada segmento une dos puntos sucesivos en los datos y se ajusta a una línea recta entre ellos. De esta manera, se logra una interpolación lineal de la función entre los puntos conocidos.

$$S_i(x) = y_i + \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \cdot (x - x_i \quad (3)$$

- Interpolacion con splines cúbicos:

$$S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \quad (4)$$

  para cada

$$j = 0, 1..., n - 1$$

  siendo n la cantidad de puntos conocidos.

  Para poder utilizar splines cubicos, se necesita que estos polinomios cumplan con ciertos criterios. A su vez, estas condiciones que deben cumplir nos permiten obtener los valores de los coeficientes a_j, b_j, c_j, d_j

  Dada una función f, S es para f una función que cumple:

  - $S(x)$ es un polinomio cúbico
  - $S(x_j) = f(x_j)$ para cada $j = 0, 1, ..., n$
  - $S_j +_1 (x_{j+1}) = S_j(x_{j+1})$

- $S'_{j+1}(x_{j+1}) = S'_j(x_{j+1})$
- $S''_{j+1}(x_{j+1}) = S''_j(x_{j+1})$

- Interpolacion con Bilineal:
  Cuando queremos hallar el valor para una función desconocida en un punto $P = (x, y)$ partimos de saber el valor de f en cuatro puntos $Q11 = (x1, y1), Q12 = (x1, y2), Q21 = (x2, y1) y Q22 = (x2, y2)$.

  Primero se hace una interpolación lineal en la dirección x. Esto genera:

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \quad (5)$$

  donde $R_1 = (x, y_1)$

$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \quad (6)$$

  Después se hace una interpolación en la dirección y:

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2) \quad (7)$$

- Interpolacion con Bicubica:

  Sea $f : R^2 \to R$ de la cuál se conoce $f(0,0)$, $f(1,0)$, $f(0,1)$, $f(1,1)$ y f_x,f_y,f_{xy}.

La superficie interpolada se puede representar como:

$$p(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j \quad (8)$$

Este polinomio cumple que:

1. $p_x(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} i x^{i-1} y^j$
2. $p_y(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^{j-1}$
3. $p_y(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i j y^{j-1}$

Entonces, p(x,y) es una superficie C^2 sobre el cuadrado $[0.1]^2$ y además, llevada a una grilla, se puede lograr que las derivadas en los bordes de cada dominio es continua.

- Metodo de Newton:
  Este metodo nos permite aproximar las raices de una función f. Se comienza con una aproximación inicial p_0 que es un punto cercano a la raiz de la función y se genera una sucesión $p_{n=0}^{\infty}$

$$p_n = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})} \qquad \text{para } n >= 1 \quad (9)$$

- Metodo de Bisección:
  Es un algoritmo utilizado para encontrar las raíces de una función continua en un intervalo dado. Dentro de un intervalo $[a, b]$ donde $f(a) \cdot f(b) < 0$ se calcula un punto c dentro del intervalo tal que $c = \frac{a+b}{2}$. Esto nos devuelve un nuevo intervalo $[a, c]$ si $f(c) \cdot f(a) < 0$ o $[c, b]$ si $f(c) \cdot f(b) < 0$. De estas primicias se deriva la fórmula:

$$p_1 = a + \frac{b - a}{2} \quad (10)$$

IV Experimentos Numéricos

IV.1 Efecto del número y posición de los puntos de interpolación

Primer experimento:
En base a los métodos previamente mencionados, llevamos a cabo una serie de experimentos numéricos con el propósito de evaluar la eficacia de cada uno y determinar los puntos donde podrían surgir dificultades en la aproximación de las funciones.

En nuestro primer experimento, nos enfocamos en comparar diferentes métodos de interpolación entre sí tomando a una función objetivo conocida. Analizamos tanto el error máximo como el error promedio, evaluando también cómo varían estos errores en función de la disposición de los puntos conocidos. Al disponer de la función original, pudimos contrastar nuestras aproximaciones y validar su precisión. Además, exploramos el efecto de variar la cantidad de puntos de interpolación. Nuestro objetivo era analizar la funcion f en el intervalo $[-4, 4]$:

$$f_a(x) = (0.3^{|x|} \cdot \sinh(4 \cdot x) - \tanh(2 \cdot x) + 2$$

Inicialmente, para llevar a cabo la interpolación de la función, decidimos tomar puntos equidistantes a lo largo de su dominio. Para esto utilizamos la librería numpy y su función linspace que divide un intervalo en una cantidad fija de puntos equidistantes.

```
inicio = -1
fin = 1
cant_puntos = 10
x = np.linspace(inicio, fin, cant_puntos)
coords_y = fa(x)
```

En nuestro primer enfoque, empleamos el polinomio de Lagrange implementado por la librería scipy de la siguente manera:

```
lagrangiano = scipy.lagrange(x, y)
```

Además variamos gradualmente la cantidad de puntos seleccionados, lo que equivale a aumentar el grado del polinomio. De esta manera, pudimos analizar cómo se comportaba y qué precisión ofrecía este método en función de la cantidad de puntos utilizados. Al tener conocimiento de la función original, pudimos calcular el error generado en los puntos que el polinomio no conocía.

Posteriormente, recurrimos a los Splines cúbicos con condición de frontera 'clamped' como segundo método de interpolación. De nuevo, nos servimos de la librería scipy:

```
splines_cúbicos = scipy.CubicSplines(x, y,
bc_type='clamped')
```

Siguiendo el mismo procedimiento de selección de puntos equidistantes, incrementamos progresivamente la cantidad de puntos conocidos por el polinomio, evaluando el error que este generaba a medida que se aumentaba el número de puntos conocidos.

Los polinomios de Lagrange formados a partir de puntos equiespaciados, cuanto mayor grado tienen, más oscilan en los bordes del intervalo, a este fenómeno se lo conoce como "fenómeno de Runge"[1]. Para contrarrestar esto, tomamos una menor cantidad de puntos en el centro y una mayor en los extremos. Esto se logró aprovechando la característica de la función coseno, que representa la proyección sobre el eje x de la circunferencia unitaria. Elegimos intervalos equidistantes entre $[\pi, 2\pi]$, lo que resultó en una menor densidad de puntos en el centro y una mayor en los bordes al aplicar la función coseno sobre estos intervalos.

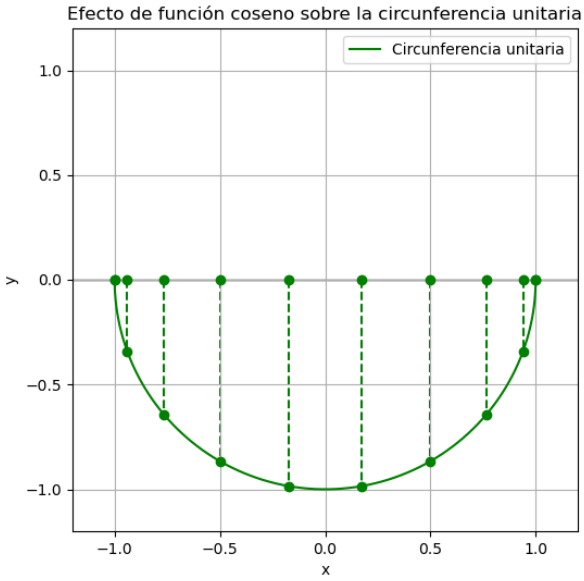

Figura 1. Puntos obtenidos para evaluar el polinomio de Lagrange

Para la interpolación con splines cúbicos, implementamos un enfoque no equiespaciado en la selección de puntos. Utilizamos la imagen de la función $\frac{1}{10} \cdot x^3 + x$ en el intervalo $[-2.478136535, 2.478136535]$. Este intervalo fue elegido estratégicamente ya que sus extremos representan los valores 4 y -4, abarcando así los límites del intervalo de interés. Además, seleccionamos esta función porque buscábamos una mayor densidad de puntos cerca del origen.

En ambos casos, nos enfocamos en calcular diversas métricas de error, tales como el error promedio, el error máximo, error sobre el dominio y el error promedio a medida que incrementamos la cantidad de puntos.

Segundo Experimento

Para nuestro segundo experimento, vamos a aplicar los métodos de interpolación a una

[1] Ver Apéndice 1

función $f : R^2 \to R$ conocida. El objetivo es analizar la eficacia de estos métodos en la interpolación de funciones en tres dimensiones.

$$f_b(x) = 0.75\left(\exp\left(-\frac{(10x_1-2)^2}{4} - \frac{(9x_2-2)^2}{4}\right)\right) + 0.65\left(\exp\left(-\frac{(9x_1+1)^2}{9} - \frac{(10x_2+1)^2}{2}\right)\right) + 0.55\left(\exp\left(-\frac{(9x_1-6)^2}{4} - \frac{(9x_2-3)^2}{4}\right)\right) - 0.01\left(\exp\left(-\frac{(9x_1-7)^2}{4} - \frac{(9x_2-3)^2}{4}\right)\right)$$

Nos centramos en comparar cómo interpolan la función f_b[2] el método de interpolación bilineal y el método de interpolación bicúbica. Es así que utilizamos la librería scipy para hacer este tipo de interpolaciones. Para la interpolación bilineal utilizamos:

```
f_interp_lineal =
scipy.interpolate(x, y, z,
kind='linear')
```

Y luego, para bicúbica, cambiamos el parámetro kind='cubic'. Nuestros x, y y z fueron construidos como una grilla de 100 puntos equiespaciados de la función original mediante la función meshgrid de la librería de python numpy:

```
x = np.linspace(-1, 1, 10)
y = np.linspace(-1, 1, 10)
z = fb(np.meshgrid(x,y))
```

Calculamos y graficamos el error promedio para ambas interpolaciones para determinar cuál de ellas se ajusta mejor a nuestra función f_b.

Posteriormente, exploramos una comparación más detallada dentro de cada método, tanto en el método bilineal como en el método bicúbico, para investigar si interpolan mejor tomando puntos equiespaciados o no. Generamos una nueva grilla, obteniendo 100 puntos de referencia. El criterio empleado consistió en calcular la imagen de la función[3] $g_1(x) = \frac{1}{10} \cdot x^3 + x$ para los valores del eje x e y en el intervalo $[-0.9216989942, 0.9216989942]$, asegurandonos de abarcar los extremos del intervalo $[-1, 1]^2$, donde se espera evaluar la función f_b. A partir del siguiente código de python obtuvimos esta grilla no equiespaciada[4]

```
def g_1(x:float) -> float:
    return x + (1/10)*np.power(x, 3)
inicio = -0.9216989942
fin = 0.9216989942
x = g_1(np.linspace(inicio, fin, 10))
y = g_1(np.linspace(inicio, fin, 10))
z = fb(np.meshgrid(x, y))
```

La función utilizada proporciona una mayor concentración de puntos cerca del origen, mientras que hacia los extremos, los puntos se dispersan más entre sí. Esta decisión se basó en el conocimiento de la naturaleza de la función, donde se observa una mayor variación cerca del origen.

Finalmente, para realizar la interpolación, los puntos seleccionados fueron las tuplas de los valores obtenidos al evaluar la función $g_1(x)$en el intervalo $[-0.9216989942, 0.9216989942]$ en el eje x y al evaluar la función $g_2(1)$ en el intervalo $[-0.9216989942, 0.9216989942]$ en el eje y. Para el eje z se evalúo la función f_b en cada uno de los puntos de la grilla y luego se realizó la interpolación como antes.

IV.2 Construcción de Trayectoria

Tercer Experimento:

En este tercer experimento, nuestro objetivo es construir una trayectoria interpolando una serie de 10 posiciones proporcionadas, para luego compararlas con la trayectoria real, la cual conoce todos los puntos por los que se pasó.

Nos encontramos con que no podíamos interpolar una trayectoria como lo hacíamos anteriormente. Entonces, lo que hicimos fue escribir a la trayectoria $\phi_1(t) = (x_1(t), y_1(t))$ y a la segunda trayectoria como $\phi_2(w) = (x_2(w), y_2(w))$. De esta forma interpolamos como antes las funciones $x_1(t)$, $x_2(w)$, $y_1(t)$ y $y_2(w)$

[2]Ver Apéndice 2
[3]Ver Apéndice 3
[4]Ver Apéndice 4

Para lograr esto, utilizamos tres métodos de interpolación: el método de interpolación de Lagrange, splines lineales y splines cúbicos, sirviéndonos, como antes, de la librería scipy. Nuestra intención es comparar el error máximo y el error promedio obtenidos por cada uno de los métodos con los 100 valores que nos sirven como ground truth. De esta manera, buscamos determinar cuál de los métodos de interpolación se ajusta mejor a nuestros datos, es decir, cuál de ellos se acerca más a la verdadera trayectoria. Como se puede ver en la siguente figura:

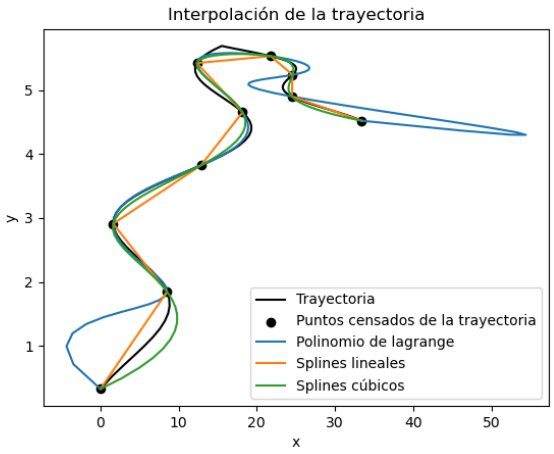

Figura 2.
Métodos de interpolación probados con la trayectoria

Cuarto Experimento:
A partir de los datos obtenidos en el experimento anterior, para interpolar esta nueva trayectoria vamos a utilizar el metodo que dio el menor error. Para calcular esta nueva trayectoria conocemos tan solo 4 puntos. Luego de interpolarla, el objetivo de este experimento será conocer en que coordenadas se cruzan ambas trayectorias. Al utilizar splines cúbicos, las coordenadas de ambas trayectorias dependen de una variable independiente cada una. Además sabemos que interpolamos con una función que genera un polinomio entre cada punto que define una de las coordenadas de la trayectoria. Entonces tomamos esos polinomios y buscamos la raíz de la función resta entre ambos. Intersecamos x_1 con x_2 y y_1 con y_2. Aproximamos la intersección mediante los métodos de búsqueda de raices de Newton-Raphson y Bisección para conocer cuál convergía con mayor velocidad con un error menor a 10^{-5}.

[5]Ver Apéndice 5

V Analisis de los resultados

A continuación abordaremos y analizaremos los resultados obtenidos de los experimentos que se mencionaron anteriormente.

V.1 Primer experimento

Durante el primer experimento, nos enfocamos en comparar dos métodos de interpolación de funciones: el método de interpolación de Lagrange y el método de interpolación con splines cúbicos.

Observamos que el error aumenta a medida que aumenta el grado del polinomio de Lagrange. Pero esta magnitud era mínima para el polinomio de grado 7, por lo que establecimos 8 como la cantidad de puntos para comparar con el método de los Splines Cúbicos.

Por otro lado, en cuanto a los splines cúbicos, observamos que a medida que incrementamos la cantidad de puntos conocidos de la función, nuestra interpolación se vuelve más precisa[5].

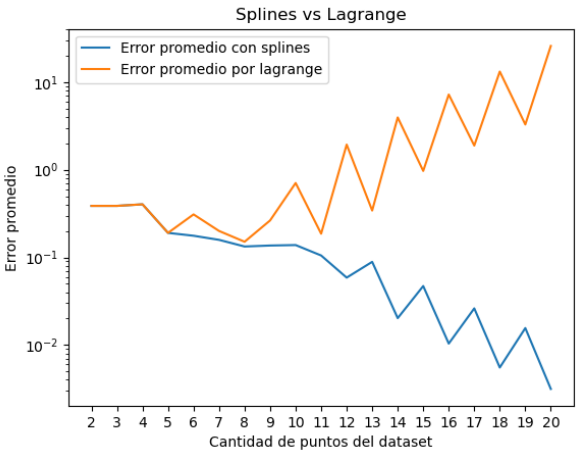

Figura 3.

Grafico del error con interpolación de Lagrange y con Splines cúbicos

Para abordar los problemas relacionados con la oscilación en los bordes del polinomio de Lagrange implementamos una modificación en la selección de puntos. Concentramos una mayor cantidad de puntos en los extremos del intervalo[6], lo que resultó en una disminución del error promedio a medida que aumentan la cantidad de puntos, como se puede ver en la siguente imágen.

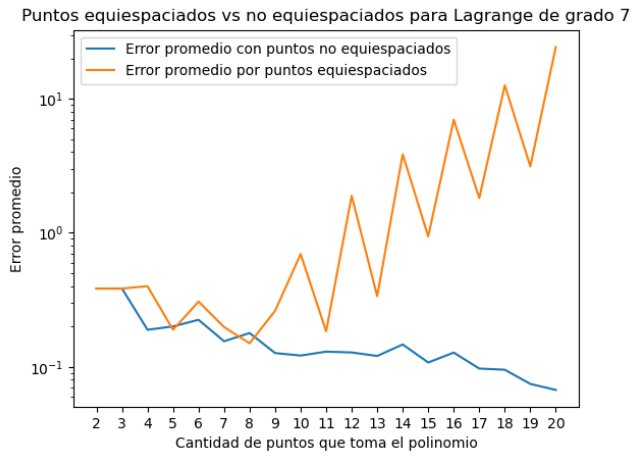

Figura 4. Grafico del error con criterios de puntos equiespaciados y no equiespaciados para la interpolación del método de Lagrange

Debido a que los splines no presentaban este tipo de inconvenientes, decidimos enfocarnos en obtener una mayor concentración de puntos cerca del origen. Sin embargo, como se puede apreciar en la Figura 5, esta estrategia no condujo a una disminución del error promedio, sino más bien a un aumento. Por lo tanto, concluimos que es más eficiente mantener una distribución equiespaciada cuando se utilizan splines cúbicos en el que caso de que se tengan pocos puntos de referencia.

No obstante, es interesante destacar que, aunque inicialmente las curvas difieren con pocos puntos conocidos, a medida que se aumenta la cantidad de puntos, ambas comienzan a mostrar similitudes hasta converger hacia un error similar. De esto podemos inferir que, si bien el criterio de selección de puntos es importante, un spline cúbico tiende a aproximar adecuadamente la función cuando se le proporciona una cantidad suficiente de puntos.

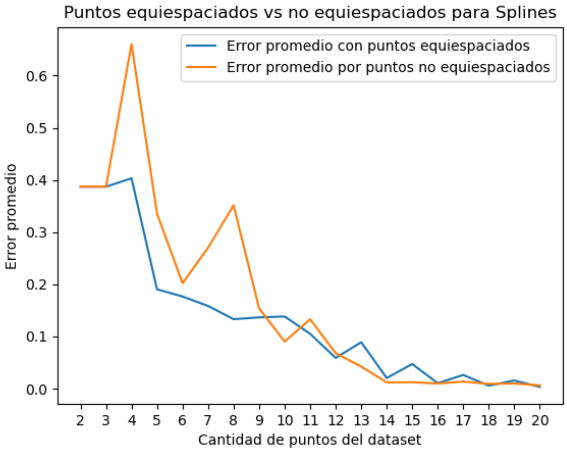

Figura 5. Grafico del error con criterios de puntos equiespaciados y no equiespaciados para la interpolación del método de Splines Cúbicos

V.2 Segundo experimento

Para el segundo experimento interpolamos nuestra función f_b con dos metodos: El metodo de interpolación bilineal y el método de interpolación bicúbica. En primera instancia queriamos saber cual de ellas interpolaba mejor a nuestra función original. De esta forma observamos que para esa cantidad de puntos equiespaciados el error promedio para la interpolación bicúbica (0.013017723087789423) era menor al de la bilineal (0.013180820571755228)[7]

Sin embargo, notamos que el error en ambas, a medida que tomabamos mas puntos se volvía decreciente. Hasta que ambas se comenzaban a estabilizar.

[6]Ver Apéndice 6
[7]Ver Apéndice 7

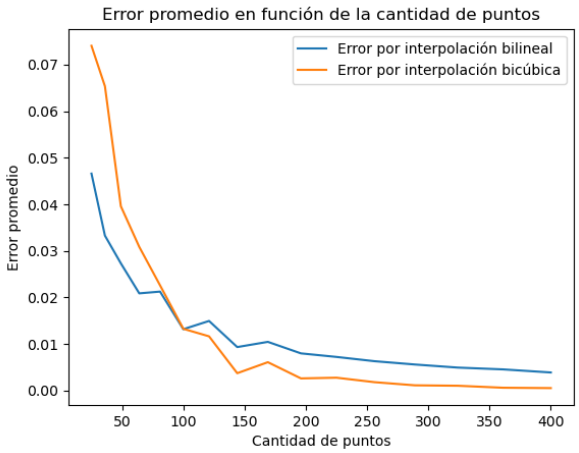

Figura 6. Comparación de errores promedio interpolación Bicúbica y Bilineal

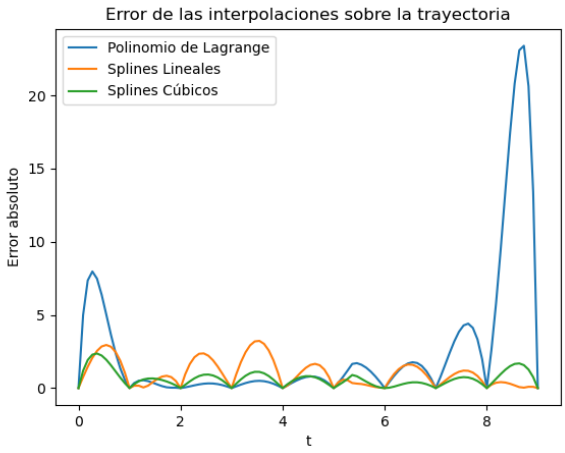

Figura 7. Comparación de error sobre la trayectoria con interpolación con polinomio de Lagrange, Splines lineales y Splines Cúbicos

Posteriormente, evaluamos si el método se volvía más efectivo al utilizar un criterio de selección de puntos equidistantes o no equidistantes[8]. Finalmente, observamos que el criterio seleccionado no mejoraba significativamente la interpolación, tanto para la Bicúbica como para la Bilineal.

Métodos	Error máximo	Error promedio
Lagrange	35,52021794	12,6166742
Splines Cúbicos	2,364684825	0,6409479368
Splines Lineales	3,227338748	1,004187489

Figura 8. Tabla comparativa del error entre los métodos que se utilizaron para interpolar una trayectoria

V.3 Tercer Experimento

Para interpolar la trayectoria, empleamos tres métodos distintos: el método de interpolación de Lagrange, el método de interpolación con splines lineales y el método de interpolación con splines cúbicos. Tras calcular sus errores promedio y maximo, llegamos a la conclusión de que los splines cúbicos ofrecían la mejor interpolación para nuestra trayectoria, como se evidencia en la siguiente figura.

[8]Ver Apéndice 8

V.4 Cuarto Experimento

Para interpolar la nueva trayectoria tomamos el método que ofreció los mejores resultados para la trayectoria anterior. Es por eso que utilizamos splines cúbicos en este caso nuevamente. Nos interesaba encontrar la intersección entre ambas trayectorias, es por esto que aprovechamos el hecho de que la interpolación con splines genera polinomios de grado 3 entre los puntos considerados. Entonces, tomamos, utilizando la librería scipy, los coeficientes de esos polinomios para las coordendadas de las trayectorias y los reconstruímos oteniendo la siguente imágen:

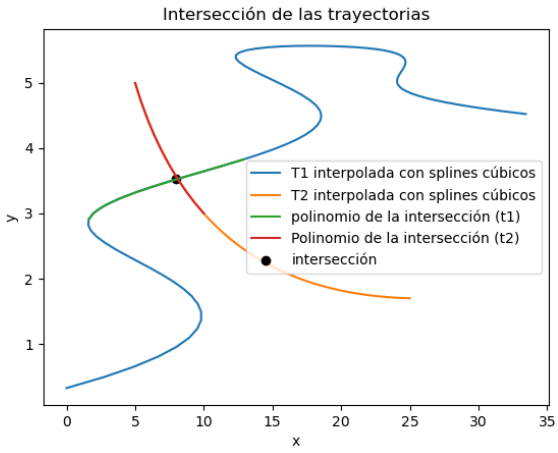

Figura 9. Polinomios que dan la intersección entre las trayectorias

A continuación, buscamos la función resta entre esos polinomios. Intersecamos coordenada a coordenada y aproximamos dicha raíz utilizando el método de Newton-Raphson y Bisección. De esta forma, observamos sus velocidades de convergencia con un error de 10^{-5} como se ve en el siguente gráfico.

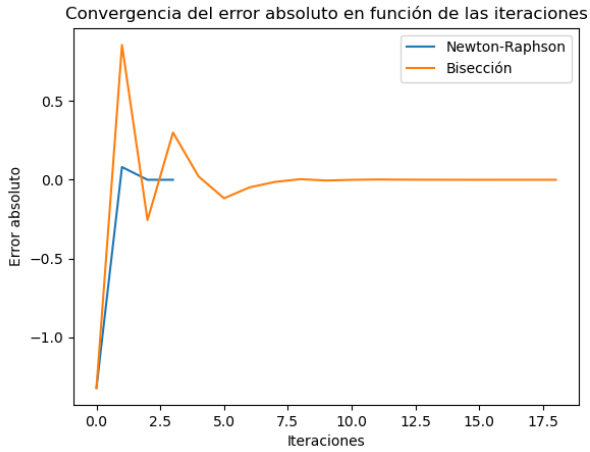

Figura 10. Convergencia de los métodos en función de las iteraiciones

VI Conclusión

En resumen, la interpolación de funciones y la búsqueda de raíces demandan una comprensión profunda de las fortalezas y debilidades inherentes a los métodos empleados. A través de nuestros experimentos, hemos evidenciado la importancia de evaluar la precisión y las limitaciones de dichos métodos. Por ejemplo, al enfrentarnos al desafío del polinomio de Lagrange, pudimos mejorar significativamente su margen de error al considerar criterios de selección de puntos no equiespaciados. No obstante, es crucial reconocer que no existe un enfoque infalible. La eficacia de un método depende de la naturaleza específica de la función original y del contexto en el que se aplique. Por tanto, al ser conscientes de estas limitaciones, podemos buscar estrategias para mitigarlas y optimizar nuestros resultados. En última instancia, la comprensión del comportamiento de estos métodos ante diversas situaciones es fundamental para abordar los desafíos de manera efectiva y obtener conclusiones precisas en nuestros análisis.

VII Apendice

VII.1 Apendice 1

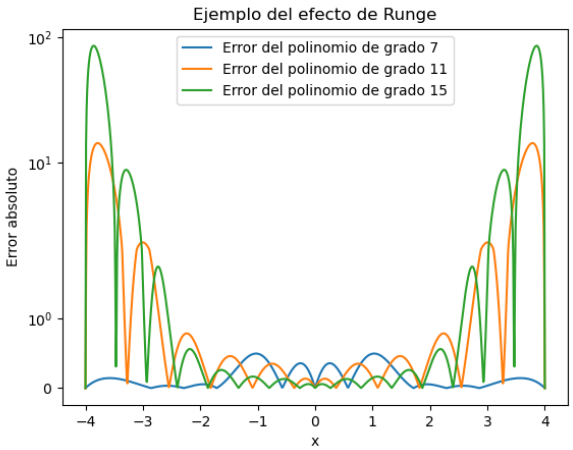

Figura 11. El error aumenta conforme se incrementa el grado del polinomio de Lagrange.

VII.2 Apendice 2

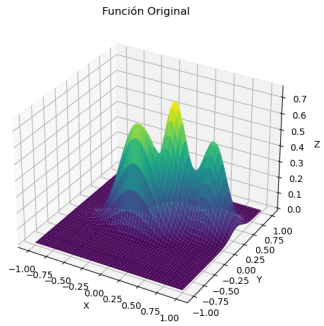

Figura 12. Grafico tridimensional de la función original

VII.3 Apendice 3

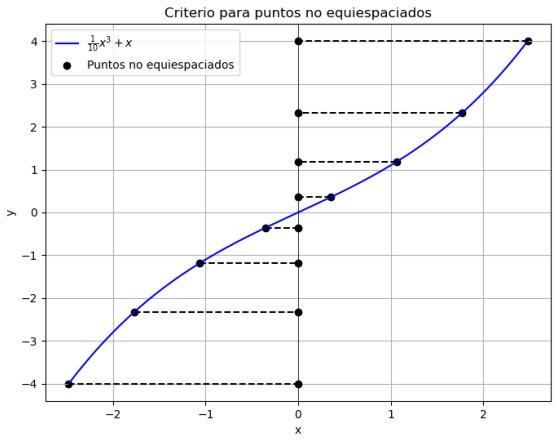

Figura 13. Puntos obtenidos de forma no equiespaciada para el metodo de bilineal y bicúbica

VII.4 Apendice 4

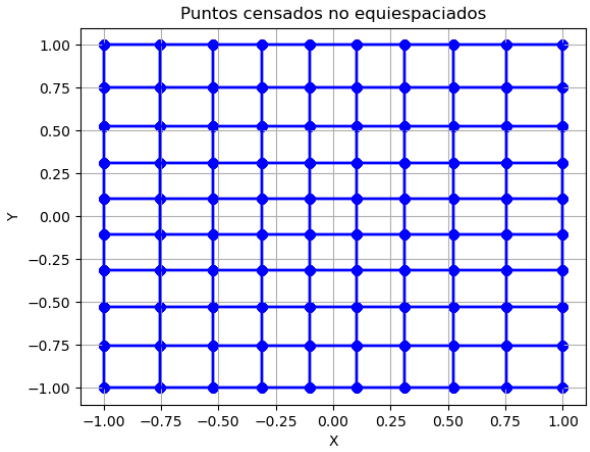

Figura 14. Grilla de puntos no equiespaciados generados de la imagen de la función $g_1(x) = \frac{1}{10} \cdot x^3 + x$.

VII.5 Apendice 5

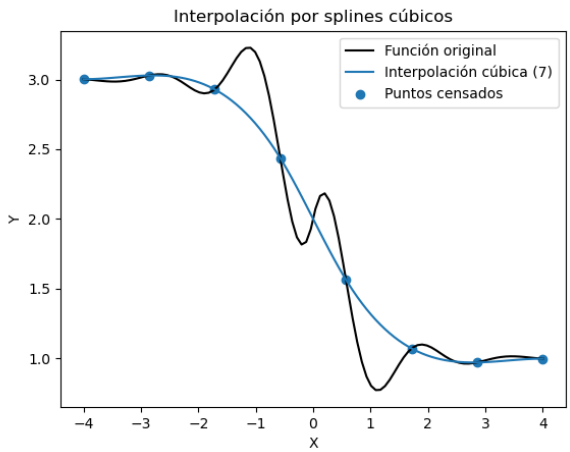

Figura 15. Mejor aproximación con Splines Cúbicos para 8 puntos de referencia

VII.6 Apendice 6

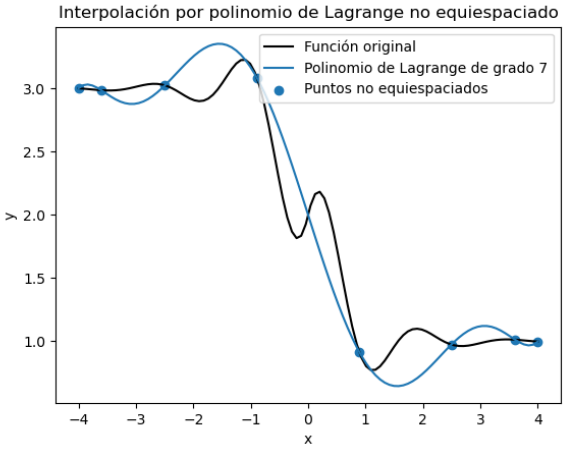

Figura 16. Mejora en la aproximación en los extremos de la función

VII.7 Apendice 7

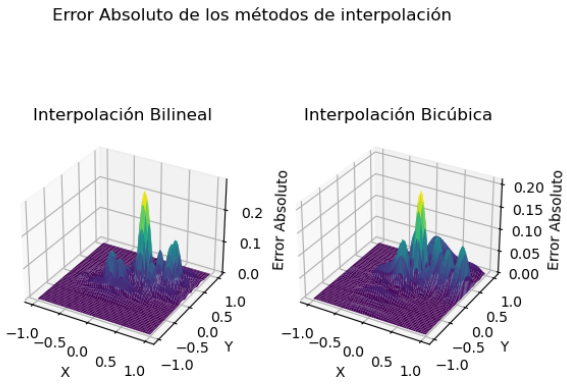

Figura 17. Grafico tridimensional del error absoluto

VII.8 Apendice 8

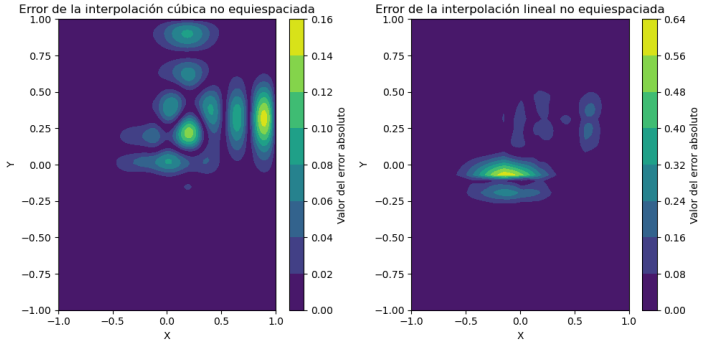

Figura 18.Error en el dominio de las interpolaciones no equidistantes.