

Capítulo 4

Interpolación

4.1 Introducción

En muchos experimentos de las ciencias básicas se obtienen puntos, por ejemplo el cambio de temperatura de un liquido, el crecimiento de una planta, el crecimiento de un ser vivo, la desintegración de un compuesto químico, todo esto en intervalos de tiempo t_0, \dots, t_n .

Estos datos por lo general se pueden graficar en un plano de coordenadas y observarse comportamientos que ya hemos observado en funciones conocidas. Veamos por ejemplo los siguientes puntos (nodos).

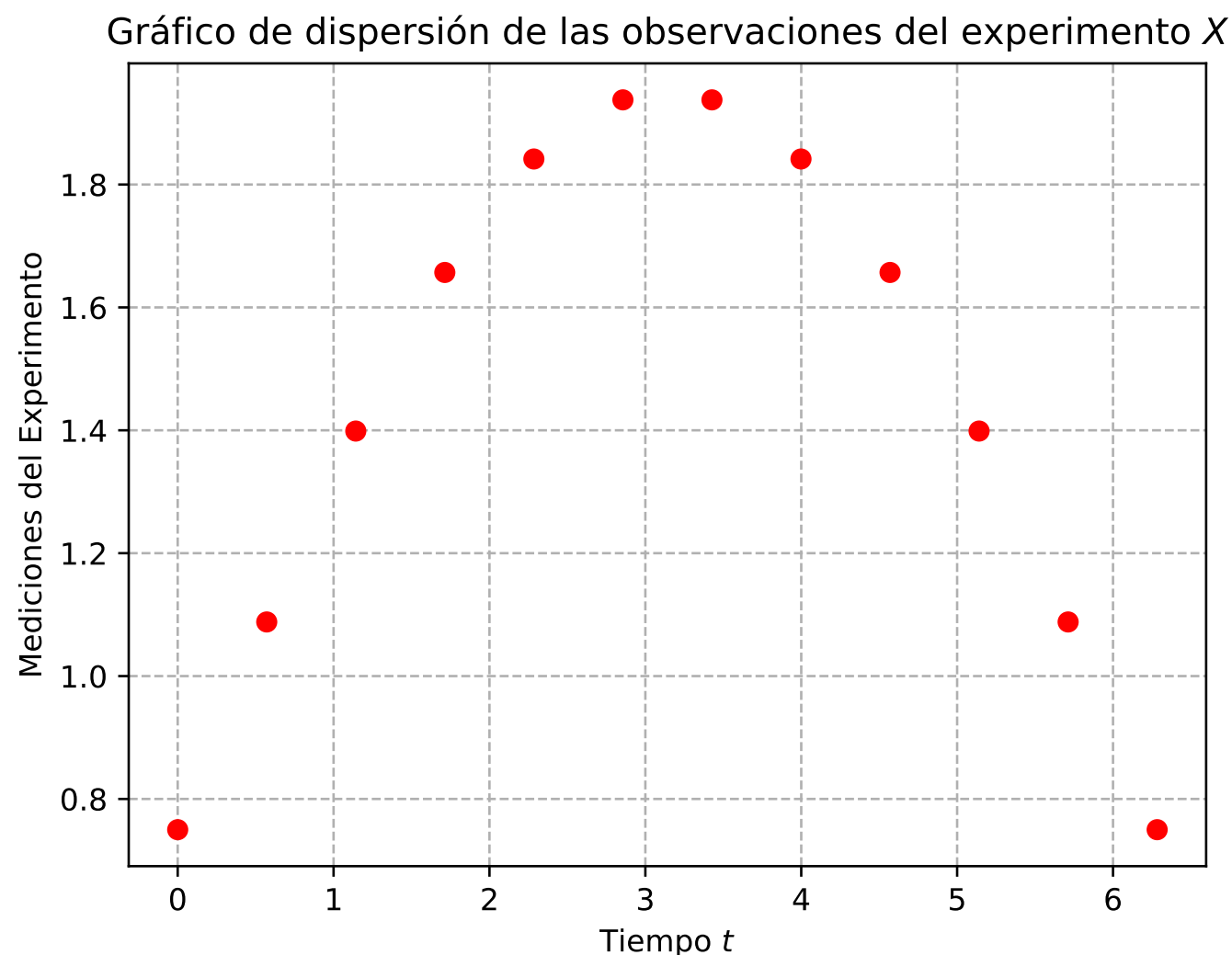


Figura 4.1: Dispersión de puntos de un experimento X

¿Qué forma tienen estos puntos?

Desde los conocimientos del cálculo diferencial estos puntos pueden representar una función polinómica. De tal manera que el objetivo de éste capítulo es estudiar cómo transformar un conjunto de puntos en un polinomio.

Sea $P_n(x)$ un polinomio de grado n , que tiene la forma

$$P_n(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 \quad (4.1)$$

que se puede representar de forma más compacta como

$$P_n(x) = \sum_{k=0}^n a_k x^k \quad (4.2)$$

4.2 Definición de Interpolación

La interpolación, en términos matemáticos formales, se define como el proceso de estimar valores desconocidos dentro de un rango discreto de puntos de datos conocidos. Formalmente:

- Dados un conjunto de puntos de datos discretos $\{(x_i, y_i)\}_{i=0}^n$, donde x_i son los valores independientes e y_i son los valores dependientes correspondientes, el objetivo de la interpolación es encontrar una función $f(x)$ que pase exactamente por todos estos puntos. Es decir, $f(x_i) = y_i$ para todo $i = 0, 1, \dots, n$.
- Esta función $f(x)$ se utiliza luego para estimar valores de y para cualquier x dentro del rango de los valores x_i (que no estaban contemplados en los datos discretos)
- **Función Interpolante:** La función $f(x)$ resultante se denomina función **interpolante**. Su forma puede variar según el método de interpolación utilizado (por ejemplo, interpolación lineal, polinómica, spline).
- **Unicidad:** La unicidad de la función interpolante depende del tipo de interpolación y de las condiciones impuestas. Por ejemplo, con $n + 1$ puntos distintos, existe un único polinomio de grado n que los interpola.
- **Aplicaciones:** La interpolación se utiliza en diversas áreas, como el análisis numérico, el procesamiento de señales, los gráficos por computadora y la estadística, para estimar valores faltantes, suavizar datos y aproximar funciones complejas.

Es importante señalar que la cantidad de nodos definen el grado de la función interpolante. En el ejemplo siguiente se observa una gráfica (figura 4.2) con cuatro (4) nodos, de tal forma que se pueden interpolar el conjunto de puntos mediante una **función cúbica**.

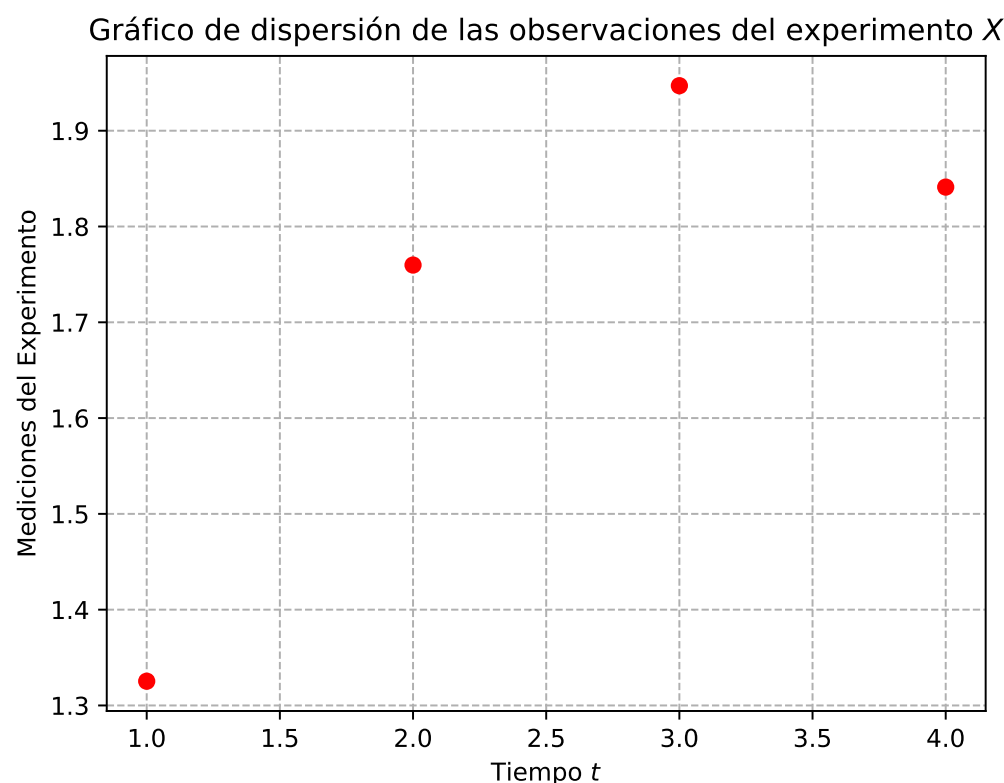


Figura 4.2: Dispersión de 4 nodos

4.3 ¿Cuál es la forma de interpolación más primitiva?

Se mencionó anteriormente que un polinomio tiene la forma

$$P_n(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

de forma particular un polinomio de grado tres (3) se puede representar de la siguiente manera

$$P_3(x) = a_3 x^3 + a_2 x^2 + a_1 x + a_0$$

Si reemplazamos un conjunto de cuatro puntos en el polinomio, vamos a obtener cuatro polinomios, que a su vez son un sistema de ecuaciones de 4×4 , veamos el siguiente ejemplo.

Ejemplo 1:

Dados el siguiente conjunto de puntos (nodos) encontrar el sistema de ecuaciones y resolver usando el módulo *linalg* de **NumPy** en **Python**.

$$\{(x_i, y_i)\}_{i=0}^3 = \{(1, 1.32), (2, 1.75), (3, 1.94), (4, 1.84)\}$$

Al reemplazar los puntos en el polinomio general de grado tres (3)

tenemos el siguiente sistema de ecuaciones

$$S = \begin{cases} a_3 + a_2 + a_1 + a_0 & = 1.32 \\ 8a_3 + 4a_2 + 2a_1 + a_0 & = 1.75 \\ 27a_3 + 9a_2 + 3a_1 + a_0 & = 1.94 \\ 64a_3 + 16a_2 + 4a_1 + a_0 & = 1.84 \end{cases}$$

que se puede representar matricialmente como

$$\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & a_3 \\ 8 & 4 & 2 & 1 & a_2 \\ 27 & 9 & 3 & 1 & a_1 \\ 64 & 16 & 4 & 1 & a_0 \end{array} \right) = \begin{pmatrix} 1.32 \\ 1.75 \\ 1.94 \\ 1.84 \end{pmatrix}$$

así se puede resolver el sistema usando el módulo **linalg** de **Numpy**, como se muestra en el siguiente código:

```

1 # Definir la matriz de coeficientes
2 A = np.array([[1, 1, 1, 1],
3               [8, 4, 2, 1],
4               [27, 9, 3, 1],
5               [64, 16, 4, 1]])
6 print("A = ", A)
7
8 # Definir el vector de términos independientes
9 b = np.array([1.32, 1.75, 1.94, 1.84])
10 print("b = ", b)
11
12 # Resolver el sistema
13 X = np.linalg.solve(A, b)
14 print("X = ", X)

```

La solución al sistema de ecuaciones es la siguiente

$$a_n = \begin{pmatrix} -0.0083 \\ -0.07 \\ 0.6983 \\ 0.7 \end{pmatrix}$$

de tal forma que el polinomio interpolador de grado tres (3) es el que se muestra a continuación

$$P_3(x) = -0.0083x^3 - 0.07x^2 + 0.6983x + 0.7$$

En la siguiente figura 4.3 se observa los nodos y el polinomio interpolador.

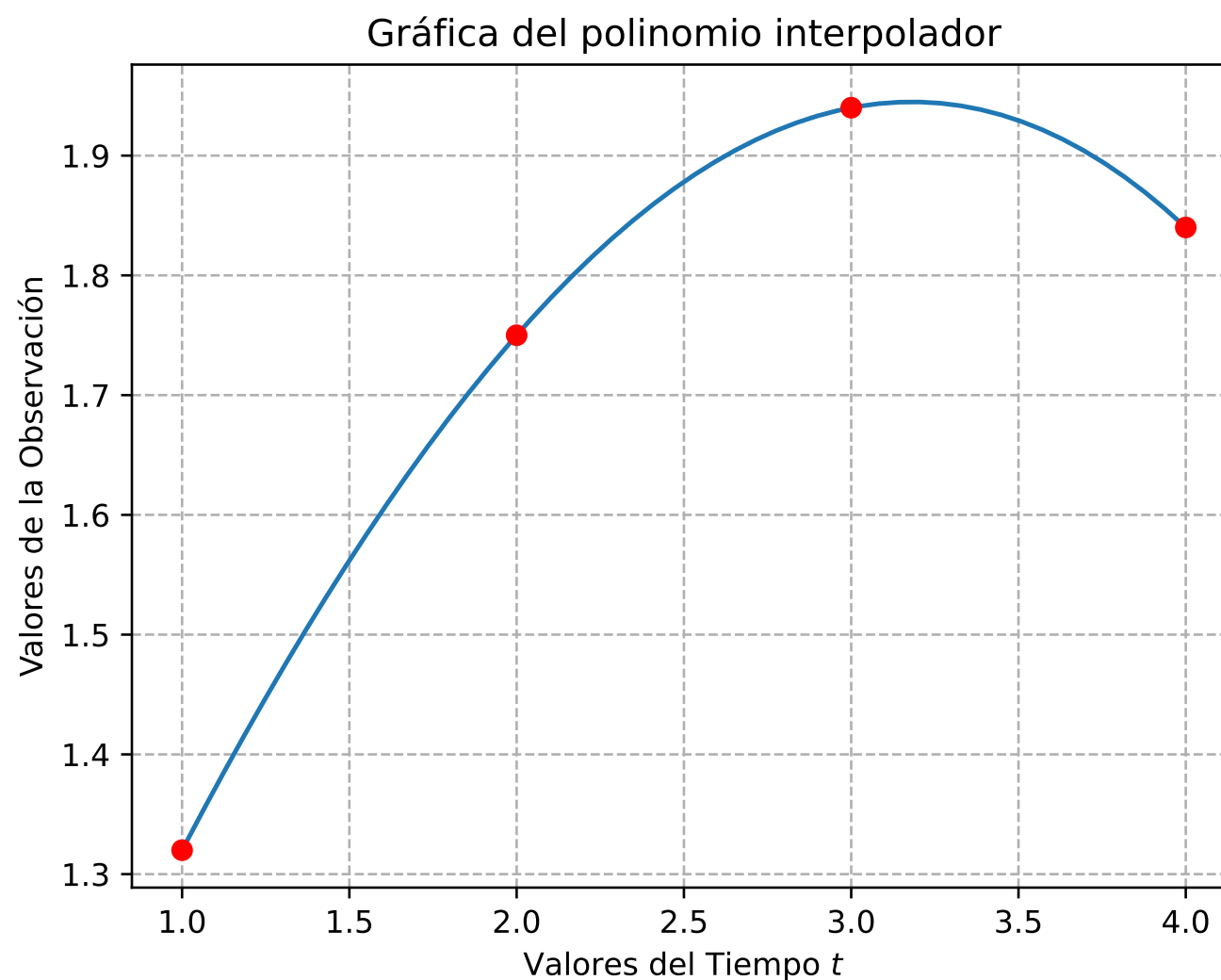


Figura 4.3: Gráfica del polinomio interpolador

Como reflexión de éste método, observamos que la solución al sistema de ecuaciones de 4×4 ha encontrado el **polinomio interpolador** que pasa por los cuatro nodos, y podemos decir lo siguiente:

- Un polinomio interpolador que pasa por $n + 1$ nodos será de grado n
- Si el polinomio a interpolar tiene pocos nodos, tal como en el ejemplo anterior, se puede resolver usando el sistema de ecuaciones.
- Si el polinomio a interpolar tiene muchos nodos hay que usar otro tipo de técnicas que no impliquen resolver sistemas de ecuaciones tan grandes.

4.4 Polinomio interpolador mediante polinomios de Lagrange

Como se observó en la sección anterior, es posible encontrar el polinomio interpolador resolviendo el sistema de ecuaciones de $n \times n$, pero no es un proceso óptimo cuando la cantidad de nodos es grande.

En esta sección introducimos el concepto de los polinomios de Lagrange.

Iniciemos con la siguiente pregunta, ¿Cómo podemos calcular un polinomio que pase exactamente por las raíces $x = x_1, x_2, \dots, x_n$?

Para ello definimos lo siguiente:

- Dado un conjunto de raíces de un polinomio $x = x_1, x_2, \dots, x_n$ el polinomio que pasa por estas raíces está dado por

$$P_n(x) = (x - x_1)(x - x_2) \cdots (x - x_n)$$

Ejemplo 2:

Sea un polinomio de grado cinco (5) que pasa por los nodos $x = 1, 2, 3, 4, 5$ encontrar su expresión algebraica

Para resolver este problema podemos hacerlo de dos formas:

- **Forma 1:** Usando productos notables, así tenemos

$$\begin{aligned} P_5(x) &= (x - 1)(x - 2)(x - 3)(x - 4)(x - 5) \\ &= (x^2 - 3x + 2)(x - 3)(x - 4)(x - 5) \\ &= (x^3 - 6x^2 + 11x - 6)(x - 4)(x - 5) \\ &= (x^4 - 10x^3 + 35x^2 - 50x + 24)(x - 5) \\ &= x^5 - 15x^4 + 85x^3 - 225x^2 + 274x - 120 \end{aligned}$$

- **Forma 2:** Usando el algoritmo de la multiplicación sintética.

Este algoritmo funciona así:

- Se inicia con el vector $[1 \ 0]$ y se multiplica por la primera raíz negativa. Al resultado se le debe insertar un cero adelante, luego

se suma el vector inicial $\begin{bmatrix} 1 & 0 \end{bmatrix}$ con el resultante $\begin{bmatrix} 0 & -1 \end{bmatrix}$, la suma de los dos vectores es $\begin{bmatrix} 1 & -1 \end{bmatrix}$

- Al vector resultante del paso anterior se multiplica por la segunda raíz negativa y se procede con los mismos pasos mencionados anteriormente, es decir:
 1. Multiplicar por la raíz negativa -2 al vector resultante del paso anterior: $-2 \begin{bmatrix} 1 & -1 \end{bmatrix} = \begin{bmatrix} -2 & 2 \end{bmatrix}$
 2. Insertamos un cero al vector del paso anterior y agregamos un cero al vector de coeficientes: $\begin{bmatrix} 0 & -2 & 2 \end{bmatrix}$ y $\begin{bmatrix} 1 & -1 & 0 \end{bmatrix}$
 3. Se suman los vectores del paso anterior: $\begin{bmatrix} 1 & -1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -2 & 2 \end{bmatrix} = \begin{bmatrix} 1 & -3 & 2 \end{bmatrix}$
- Se realiza el proceso iterativo hasta terminar con todas las raíces.

Δ Como se observa en la siguiente tabla 4.1, se muestra todo el proceso iterativo. La clave fundamental de este algoritmo es siempre **insertar un cero** al resultado del primer producto y **agregar un cero** al vector de inicio para que la suma sea dimensionalmente posible.

1	0					
0	-1					$x - 1$
1	-1	0				
0	-2	2				$x - 2$
1	-3	2	0			
0	-3	9	-6			$x - 3$
1	-6	11	-6	0		
0	-4	24	-44	24		$x - 4$
1	-10	35	-50	24	0	
0	-5	50	-175	250	-120	$x - 5$
1	-15	85	-225	274	-120	

Cuadro 4.1: Proceso iterativo de la multiplicación sintética

- ❑ Este proceso se puede sintetizar en un código como se muestra a continuación.

4.4.1 Multiplicación Sintética

A continuación se presenta el *código* en **Python** para encontrar los coeficientes de un polinomio de grado n dadas n raíces.

```

1 # Definir el método de Multiplicación Sintética
2 def MutiplicacionSintetica(raices):
3
4     # Crear el vector inicial
5     Coeficientes = np.array([1, 0])
6
7     # Crear un ciclo para recorrer las raíces
8     for r in raices:
9
10        # Multiplicar el vector de coeficientes por la
11        raíz negativa
12        Factor = -r * Coeficientes
13
14        # Insertar un cero al Factor
15        Factor = np.insert(Factor, 0, 0)
16        Factor = Factor[:-1]
17
18        # Sumar los vectores y actualizar los
19        coeficientes
20        Coeficientes = Coeficientes + Factor
21
22        # Agregar un nuevo cero a la lista de
23        coeficientes
24        Coeficientes = np.append(Coeficientes, 0)
25
26    # Devolver el vector de coeficientes
27    return Coeficientes[:-1]

```

Como ejemplo se ejecutan las siguientes líneas

```

1 # Definir el conjunto de raíces
2 raices = np.array([1, 2, 3, 4, 5])
3
4 # Calcular los coeficientes

```

```
5 Coeficientes = MultiplicacionSintetica(raices)
6 print(Coeficientes)
```

El resultado es el siguiente (coeficientes del polinomio de grado cinco (5))

$$[1 \quad -15 \quad 85 \quad -225 \quad 274 \quad -120]$$

Ahora la pregunta es, ¿Cómo podemos evaluar algún valor x para este polinomio encontrado?

4.4.2 Evaluación de Polinomios

Como se mencionó en la ecuación 4.1 tiene su forma compacta 4.2 siguiente

$$P_n(x) = \sum_{k=0}^n a_k x^k$$

y como tenemos ahora representado mediante el algoritmo de la **Multiplicación Sintética** los coeficientes del polinomio en un vector, entonces esta forma compacta tiene mucho sentido. A continuación se muestra un *código* de **Python** que nos permite evaluar una función polinómica dados los coeficientes y el valor que se desea evaluar.

```
1 # Definimos la función evalPoly
2 def evalPoly(coeficientes, x):
3
4     # Iniciar la suma en cero
5     suma = 0
6
7     # Determinar el grado de la función polinómica
8     n = len(coeficientes) - 1
9
10    # Crear un ciclo para recorrer los coeficientes del
        polinomio
11    for k, a in enumerate(coeficientes):
12
13        # Calcular el valor de la serie
14        suma += a * x**(n-k)
15
```

```

16 # Devolver la evaluación
17 return suma

```

Esta función nos permite por ejemplo obtener puntos específicos del polinomio, inclusive realizar la gráfica. A continuación se presenta un ejemplo de ello.

Ejemplo 3:

Encontrar mediante el algoritmo de la **Multiplicación Sintética** un polinomio que pase por las raíces

$$x = 2, 3, 5, 7, 11$$

Luego verificar la validez del polinomio evaluando el conjunto de raíces mediante la función **evalPoly** y realizar una gráfica de la función polinómica en el intervalo cerrado $[2, 11]$

Aplicando el algoritmo de la Multiplicación Sintética se tiene el siguiente resultado

$$\{a_{5-k}\}_{k=0}^5 = [1 \quad -28 \quad 288 \quad -1358 \quad 2927 \quad -2310]$$

La gráfica de la función polinómica de grado cinco (5),

$$P_5(x) = x^5 - 28x^4 + 288x^3 - 1358x^2 + 2927x - 2310$$

es la que se muestra en la siguiente figura

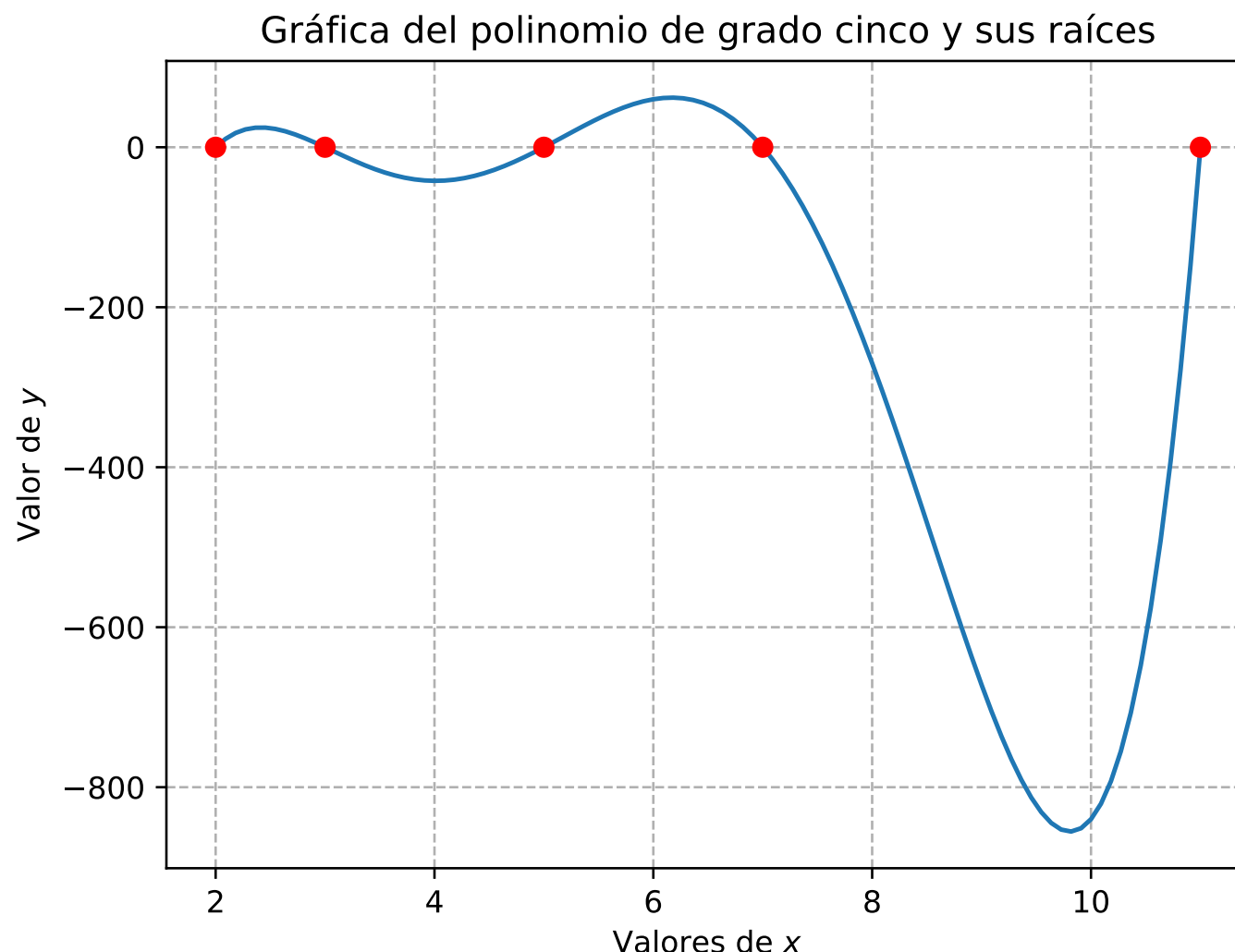


Figura 4.4: Gráfica de la función polinómica $P_5(x) = x^5 - 28x^4 + 288x^3 - 1358x^2 + 2927x - 2310$ y las raíces

4.4.3 Polinomios de Lagrange

Ya tenemos una base para calcular polinomios de grado n a través del algoritmo de la Multiplicación Sintética, ahora definimos un polinomio de Lagrange, como el polinomio que pasa por $n - 1$ nodos excepto en el nodo k . Sean el conjunto de nodos $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ se define un polinomio de **Lagrange** de grado n del nodo k como

$$L_{n,k}(x) = \frac{(x - x_0)(x - x_1) \cdots (x - x_{k-1})(x - x_{k+1}) \cdots (x - x_n)}{(x_k - x_0)(x_k - x_1) \cdots (x_k - x_{k-1})(x_k - x_{k+1}) \cdots (x_k - x_n)} \quad (4.3)$$

es un polinomio que si se evalúa en cualquier nodo x_i da como resultado cero (0) pero si se evalúa en el nodo x_k da como resultado uno (1).

Veamos el siguiente ejemplo:

Ejemplo 4:

Consideremos los siguientes nodos de observación

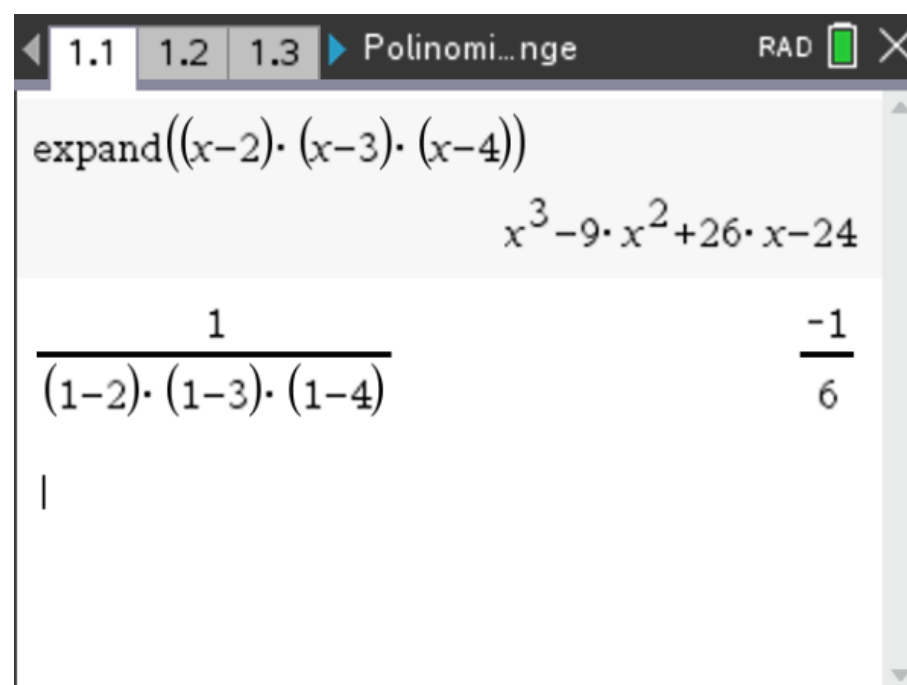
$$(1, 2), (2, 3), (3, 5), (4, 7)$$

y calculemos los polinomios de Lagrange para cada nodo

Tenemos los siguientes polinomios de Lagrange en base a la ecuación 4.3:

○ $L_{3,0}(x)$ es

$$\begin{aligned} L_{3,0}(x) &= \frac{(x-2)(x-3)(x-4)}{(1-2)(1-3)(1-4)} \\ &= -\frac{1}{6}(x^3 - 9x^2 + 26x - 24) \end{aligned}$$



The screenshot shows a calculator interface with the following content:

- Top bar: 1.1, 1.2, 1.3, Polinomi...nge, RAD, battery icon, close button.
- Input field: $\text{expand}((x-2) \cdot (x-3) \cdot (x-4))$
- Result field: $x^3 - 9 \cdot x^2 + 26 \cdot x - 24$
- Below the result, a fraction is shown: $\frac{1}{(1-2) \cdot (1-3) \cdot (1-4)}$
- To the right of the fraction, the value $-\frac{1}{6}$ is displayed.

○ $L_{3,1}(x)$ es

$$\begin{aligned} L_{3,1}(x) &= \frac{(x-1)(x-3)(x-4)}{(2-1)(2-3)(2-4)} \\ &= \frac{1}{2}(x^3 - 8x^2 + 19x - 12) \end{aligned}$$

The screenshot shows a calculator window titled '*Polinomi...nge' with tabs 1.1 and 1.2. The input is $\text{expand}((x-1) \cdot (x-3) \cdot (x-4))$ and the output is $x^3 - 8 \cdot x^2 + 19 \cdot x - 12$. Below this, the fraction $\frac{1}{(2-1) \cdot (2-3) \cdot (2-4)}$ is shown, with the result $\frac{1}{2}$.

○ $L_{3,2}(x)$ es

$$\begin{aligned} L_{3,2}(x) &= \frac{(x-1)(x-2)(x-4)}{(3-1)(3-2)(3-4)} \\ &= -\frac{1}{2}(x^3 - 7x^2 + 14x - 8) \end{aligned}$$

The screenshot shows a calculator window titled 'Polinomi...nge' with tabs 1.2, 1.3, and 1.4. The input is $\text{expand}((x-1) \cdot (x-2) \cdot (x-4))$ and the output is $x^3 - 7 \cdot x^2 + 14 \cdot x - 8$. Below this, the fraction $\frac{1}{(3-1) \cdot (3-2) \cdot (3-4)}$ is shown, with the result $\frac{-1}{2}$.

○ y $L_{3,3}(x)$ es

$$\begin{aligned} L_{3,3}(x) &= \frac{(x-1)(x-2)(x-3)}{(4-1)(4-2)(4-3)} \\ &= \frac{1}{6}(x^3 - 6x^2 + 11x - 6) \end{aligned}$$

The screenshot shows a calculator window titled "Polinomi...nge" with a "RAD" indicator. The input field contains the expression $\text{expand}((x-1) \cdot (x-2) \cdot (x-3))$. The result displayed is $x^3 - 6 \cdot x^2 + 11 \cdot x - 6$. Below this, the fraction $\frac{1}{(4-1) \cdot (4-2) \cdot (4-3)}$ is shown, with the simplified result $\frac{1}{6}$ displayed to the right.

Un ejercicio adicional consiste en graficar estos cuatro polinomios resultantes y ver como cada uno pasa por tres de las raíces (nodos) excepto en la del nodo k propuesto, veamos la gráfica en la siguiente figura 4.5.

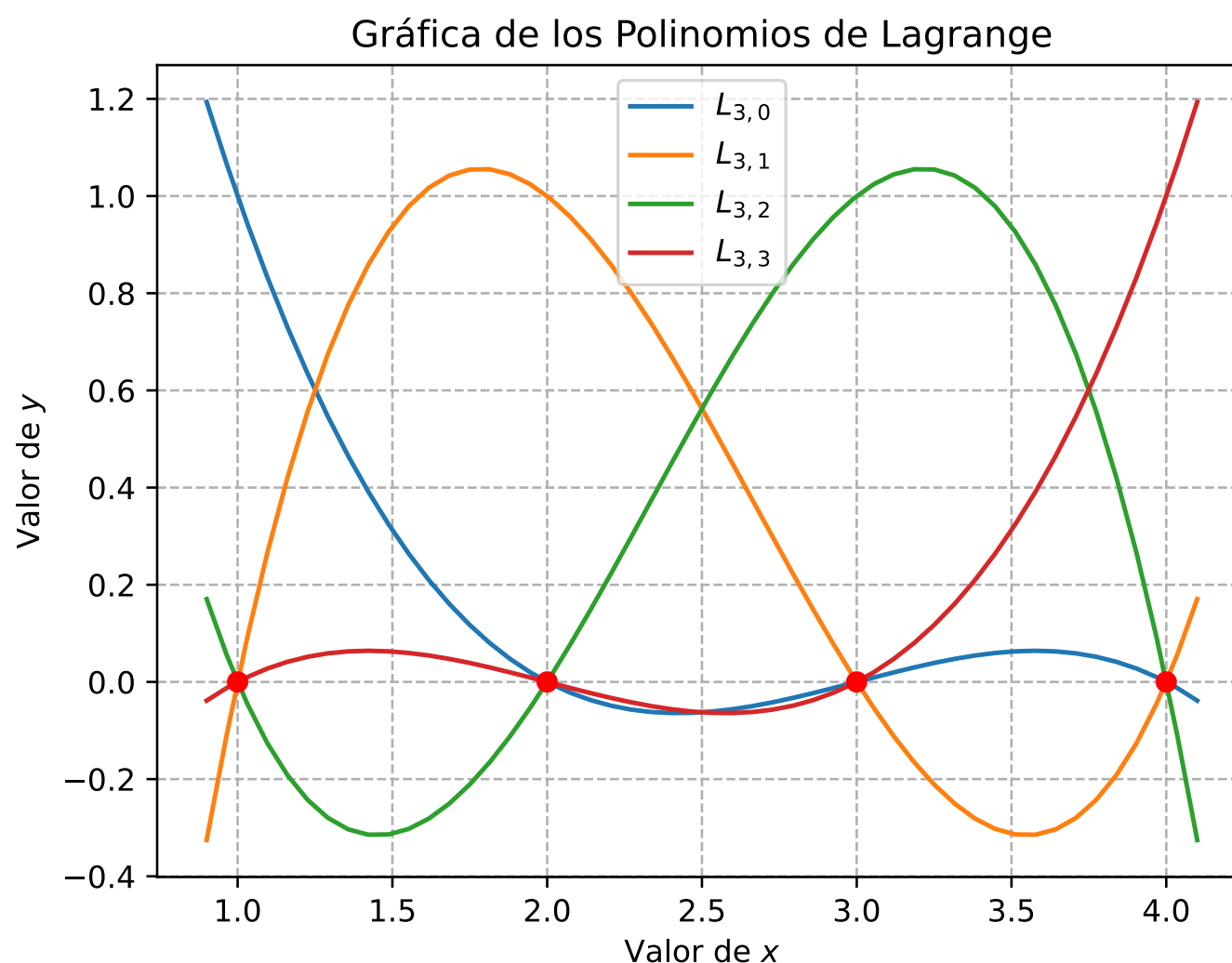


Figura 4.5: Gráfica de los Polinomios de Lagrange

Esta gráfica se consigue a través del siguiente código:

```
1 # Definir cada uno de los polinomios
2 def L_3_0(x):
```

```
3     return (-1/6)*(x**3-9*x**2+26*x-24)
4
5 def L_3_1(x):
6     return (1/2)*(x**3-8*x**2+19*x-12)
7
8 def L_3_2(x):
9     return (-1/2)*(x**3-7*x**2+14*x-8)
10
11 def L_3_3(x):
12     return (1/6)*(x**3-6*x**2+11*x-6)
13
14 # Crear el dominio de graficación
15 x = np.linspace(0.9,4.1,50)
16
17 # Evaluar cada polinomio
18 y_L_3_0 = L_3_0(x)
19 y_L_3_1 = L_3_1(x)
20 y_L_3_2 = L_3_2(x)
21 y_L_3_3 = L_3_3(x)
22
23 # Crear los nodos
24 nodos_x = np.array([1, 2, 3, 4])
25 nodos_y = np.array([0, 0, 0, 0])
26
27 # Graficar los polinomios y las raíces
28 plt.plot(x, y_L_3_0)
29 plt.plot(x, y_L_3_1)
30 plt.plot(x, y_L_3_2)
31 plt.plot(x, y_L_3_3)
32
33 plt.legend(["$L_{3,0}$", "$L_{3,1}$", "$L_{3,2}$", "$L_{3,3}$"])
34
35 plt.scatter(nodos_x, nodos_y, c='red', zorder=6)
36
37 plt.grid(linestyle='—')
38 plt.xlabel("Valor de $x$")
```



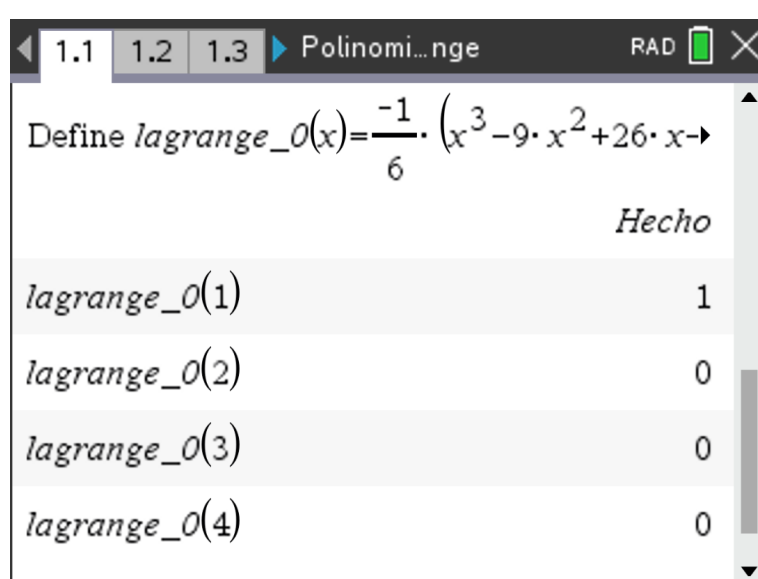
```

39 plt.ylabel("Valor de $y$")
40 plt.title("Gráfica de los Polinomios de Lagrange")
41
42 plt.savefig("../Interpolacion-Aproximacion-Polinomial/
    Imagenes/Polinomios-Lagrange-Grado-3.pdf")
43 plt.show()

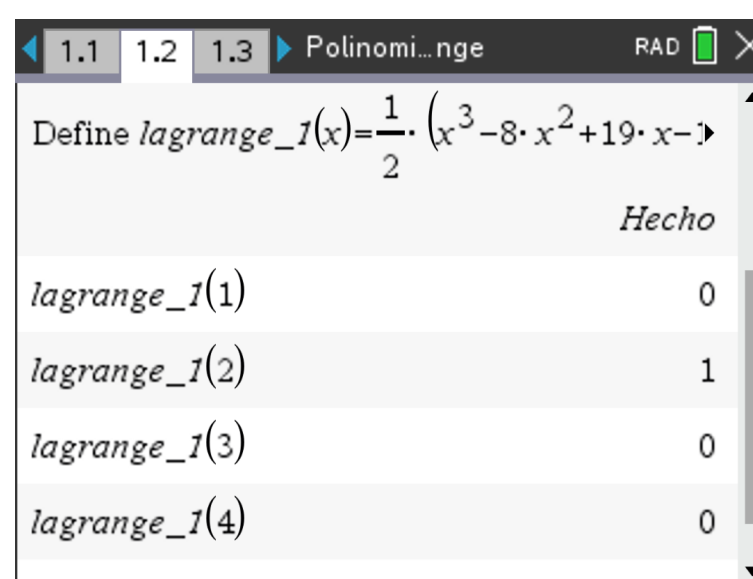
```

4.4.4 Polinomio Interpolador Mediante Lagrange

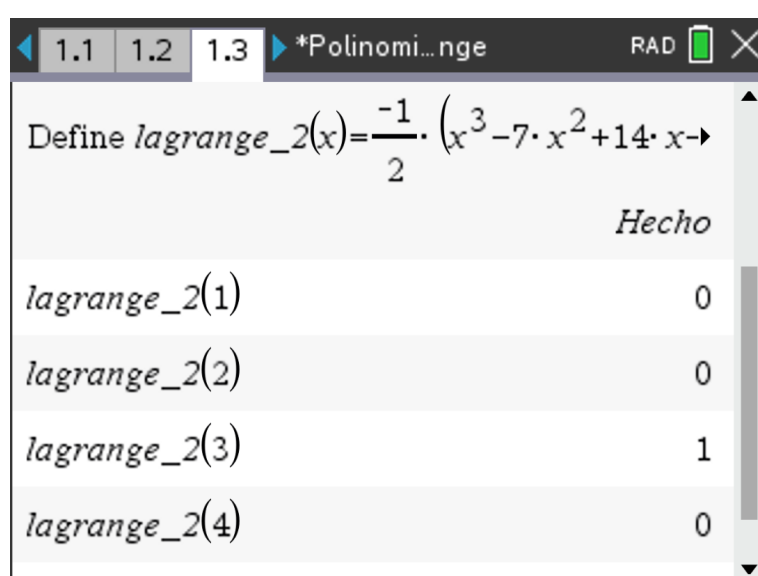
Como se mencionó anteriormente, cada uno de los polinomios de Lagrange dan como resultado cero (0) si se evalúan para algún x_i con $i \neq k$, es decir que si tenemos $L_{n,k}(x_i)_{i \neq k} = 0$ y es uno (1) si se evalúa en el nodo x_k , es decir $L_{n,k}(x_k) = 1$ veamos los siguientes cálculos



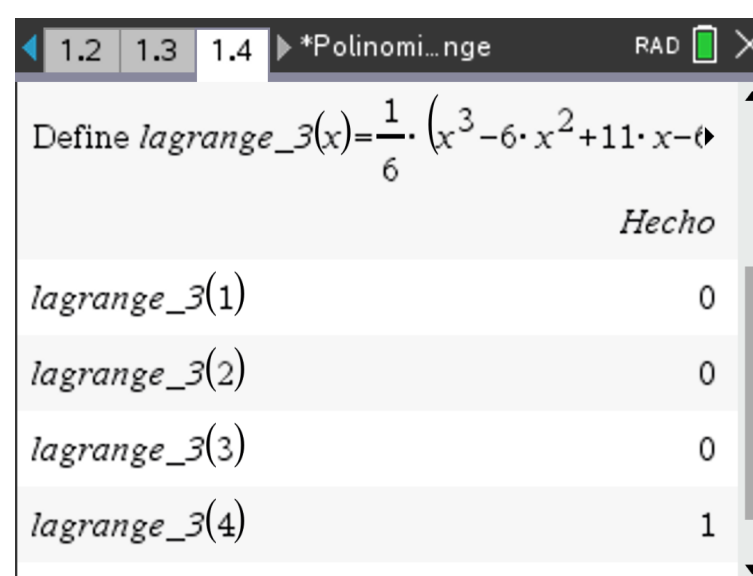
(a) Evaluación de $L_{3,0}(x)$



(b) Evaluación de $L_{3,1}(x)$



(c) Evaluación de $L_{3,2}(x)$



(d) Evaluación de $L_{3,3}(x)$

Figura 4.6: Evaluación de los Polinomios de Lagrange

Cuando se evalúa cada polinomio en el valor k donde fue definido obtenemos como resultado uno (1). Esto permite crear un polinomio

interpolador que pase exactamente por cada nodo.

○ Para el nodo $(x_0, y_0) = (1, 2)$ se definió el polinomio interpolador

$$L_{3,0}(x) = -\frac{1}{6}(x^3 - 9x^2 + 26x - 24)$$

que al evaluarlo en cualquier nodo diferente de x_0 da como resultado cero (0) pero al evaluarlo en $x_0 = 1$ da como resultado uno (1), si lo multiplicamos por el valor de $y_0 = 2$ obtenemos un polinomio que al evaluarlo en $x_0 = 1$ da como resultado el valor esperado $y_0 = 2$, así

$$y_0 L_{3,0}(x) = -\frac{1}{3}(x^3 - 9x^2 + 26x - 24)$$

Este mismo proceso se aplica para los otros nodos.

○ Para el nodo $(x_1, y_1) = (2, 3)$ tenemos

$$\begin{aligned} y_1 L_{3,1}(x) &= 3 \left[\frac{1}{2}(x^3 - 8x^2 + 19x - 12) \right] \\ &= \frac{3}{2}(x^3 - 8x^2 + 19x - 12) \end{aligned}$$

○ Para el nodo $(x_2, y_2) = (3, 5)$ tenemos

$$\begin{aligned} y_2 L_{3,2}(x) &= 5 \left[-\frac{1}{2}(x^3 - 7x^2 + 14x - 8) \right] \\ &= -\frac{5}{2}(x^3 - 7x^2 + 14x - 8) \end{aligned}$$

○ y Para el nodo $(x_3, y_3) = (4, 7)$ tenemos,

$$\begin{aligned} y_3 L_{3,3}(x) &= 7 \left[\frac{1}{6}(x^3 - 6x^2 + 11x - 6) \right] \\ &= \frac{7}{6}(x^3 - 6x^2 + 11x - 6) \end{aligned}$$

Ejercicio 1:

Construir un *programa* en **Python** que permita calcular los polinomios de Lagrange de forma automática

A continuación se presenta el código para calcular los polinomios de Lagrange dado el conjunto de nodos.

```

1 # Definir la función que calcula el polinomio de
  Lagrange
2 def polyLagrange(nodos, k):
3
4     # Crear un vector de nodos excepto el k
5     headNodos = nodos[:k]
6     tailNodos = nodos[k+1:]
7     nodos_ = np.concatenate((headNodos, tailNodos))
8
9     # Calcular la cantidad de nodos
10    n = len(nodos_)
11
12    # Calcular el denominador
13    k_ = nodos[k]
14    denominador = 1
15    for i in range(n):
16        denominador = denominador * (k_ - nodos_[i])
17
18    # Usar la función de Multiplicación Sintética para
      calcular el polinomio
19    return MultiplicacionSintetica(nodos_) / denominador

```

Una prueba del cálculo de los polinomios se observa en el siguiente código

```

1 # Definir el conjunto de nodos
2 x = np.array([1, 2, 3, 4])
3
4 L_3_0 = polyLagrange(x, 0)
5 print(L_3_0)
6
7 L_3_1 = polyLagrange(x, 1)
8 print(L_3_1)
9
10 L_3_2 = polyLagrange(x, 2)
11 print(L_3_2)
12
13 L_3_3 = polyLagrange(x, 3)

```

14 **print**(L_3_3)

El resultado de la ejecución es la siguiente

$$\begin{aligned} L_{3,0} &= [-0.16666667 \ 1.5 \ -4.33333333 \ 4.] \\ L_{3,1} &= [0.5 \ -4. \ 9.5 \ -6.] \\ L_{3,2} &= [-0.5 \ 3.5 \ -7. \ 4.] \\ L_{3,3} &= [0.16666667 \ -1. \ 1.83333333 \ -1] \end{aligned}$$

Ahora, como todos los polinomios de Lagrange son linealmente independientes dado que pertenecen a cada nodo k entonces el polinomio interpolador esta dado por

$$P_n(x) = \sum_{k=0}^n y_k \cdot L_{n,k}(x)$$

Para el ejemplo que tenemos

$$\begin{aligned} P_3(x) &= -\frac{1}{3}(x^3 - 9x^2 + 26x - 24) + \frac{3}{2}(x^3 - 8x^2 + 19x - 12) \\ &\quad - \frac{5}{2}(x^3 - 7x^2 + 14x - 8) + \frac{7}{6}(x^3 - 6x^2 + 11x - 6) \\ &= -\frac{1}{6}(x^3 - 9x^2 + 14x - 18) \end{aligned}$$

Ejercicio 2:

Crear un *programa* en **Python** que permita graficar el polinomio interpolador

$$P_3(x) = -\frac{1}{6}(x^3 - 9x^2 + 14x - 18)$$

y mostrar los nodos $(1, 2)$, $(2, 3)$, $(3, 5)$ y $(4, 7)$ sobre la gráfica

El código que soluciona este ejercicio es el siguiente:

```
1 # Definir al polinomio interpolador
2 def P3(x):
3     return (-1/6)*(x**3-9*x**2+14*x-18)
4
5 # Definir el dominio de graficación
6 x = np.linspace(0.9, 4.1, 100)
7
```

```
8 # Evaluar el polinomio
9 y = P3(x)
10
11 # Definir los nodos
12 nodos_x = np.array([1, 2, 3, 4])
13 nodos_y = np.array([2, 3, 5, 7])
14
15 # Graficar la función y los nodos
16 plt.plot(x, y)
17 plt.scatter(nodos_x, nodos_y, c='red', zorder=3)
18 plt.grid(linestyle='—')
19 plt.xlabel("Valores de  $x$ ")
20 plt.ylabel("Valores de  $y$ ")
21 plt.title("Gráfica del Polinomio Interpolador y los Nodos")
22 plt.savefig('../Interpolacion-Aproximacion-Polinomial/
    Imagenes/polinomio-interpolador-mediante-lagrange.pdf')
23 plt.show()
```

El gráfico del polinomio interpolador y los nodos se observa en la siguiente figura 4.7

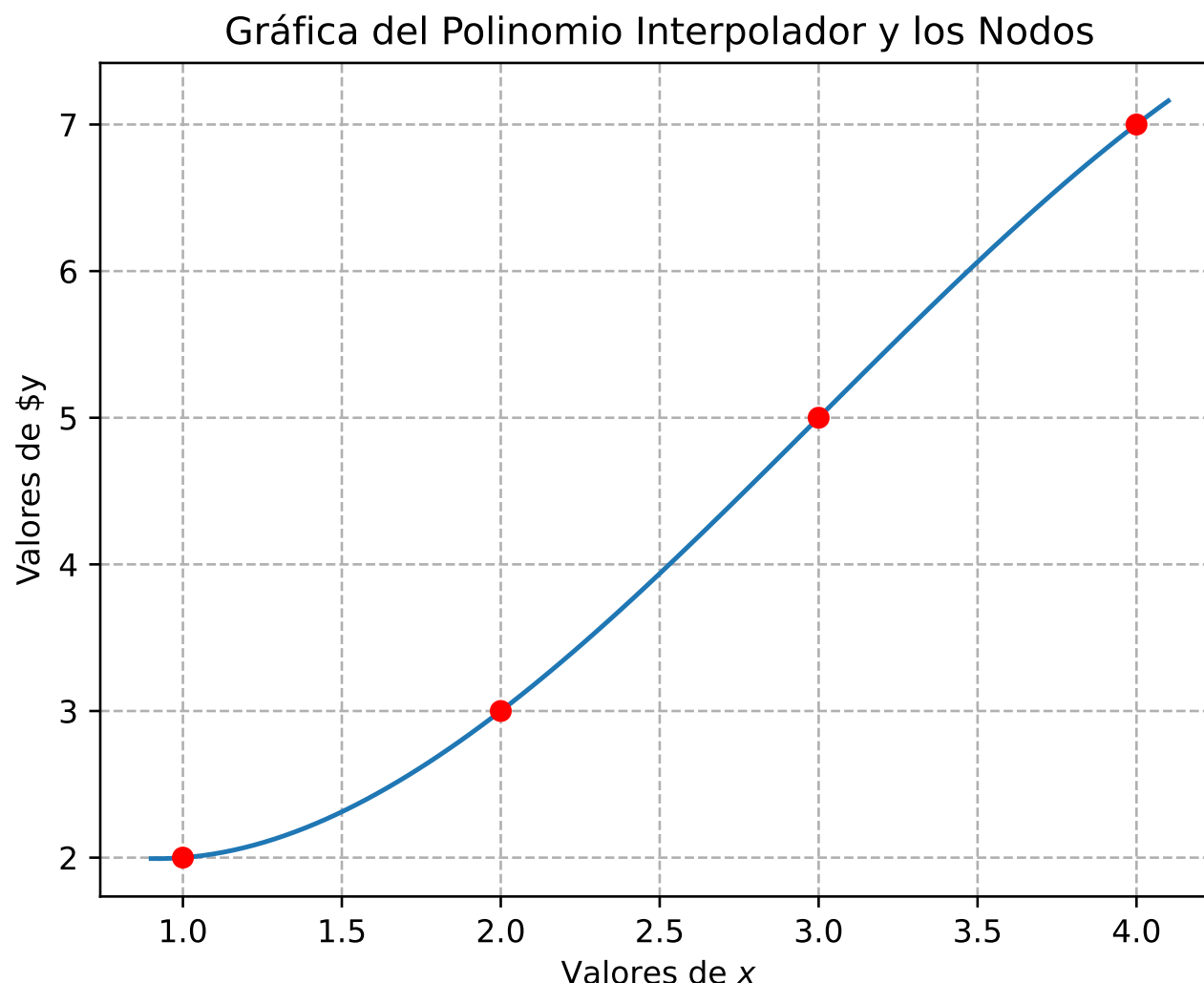


Figura 4.7: Gráfica del polinomio interpolador $P_3(x) = -\frac{1}{6}(x^3 - 9x^2 + 14x - 18)$

Algunas conclusiones sobre este ejercicio son:

- En el contexto real, vamos a requerir calcular polinomios más grandes, así que realizar los cálculos de forma manual requerirá gran esfuerzo.
- Para optimizar los cálculos es necesario implementar un código que permita optimizar estos procesos.

4.4.5 Código para el polinomio interpolador usando Lagrange

A continuación se presentan los pasos para crear el código:

1. Obtener los vectores de x e y
2. Obtener la cantidad de nodos n
3. Crear un ciclo para calcular cada polinomio de Lagrange

El código es el siguiente:

```

1 # Definir la funcion
2 def polyInterpoLagrange(nodos):
3
4     # Obtener los vectores de x e y
5     x, y = nodos
6
7     # Obtener la cantidad de nodos
8     n = len(x)
9
10    # Crear un ciclo para calcular cada polinomio de
        Lagrange
11    polyInterpola = np.zeros(n)
12
13    for k in range(n):
14        polyInterpola += y[k] * polyLagrange(x, k)
15
16    # Devolver el resultado
17    return polyInterpola

```

Para usar el código con el ejemplo anterior tenemos:

```

1 # Definir los nodos
2 nodos_x = [1, 2, 3, 4]
3 nodos_y = [2, 3, 5, 7]
4 nodos = (nodos_x, nodos_y)
5
6 # Usar la función de interpolación
7 poly = polyInterpoLagrange(nodos)
8 print(poly)

```

El resultado del polinomio interpolador es

$$P_3(x) = [-0.16666667 \ 1.5 \ -2.33333333 \ 3.]$$

es decir

$$P_3(x) = -\frac{1}{6}(x^3 - 9x^2 + 14x - 18)$$

Ejercicio 3:

Calcular un polinomio interpolador usando polinomios de Lagrange para los números primos del 2 al 100. Indexe cada número primo con los números naturales.

Este ejercicio es interesante ya que nos permite escribir un fragmento de código para obtener los números primos en el intervalo de 2 a 100, tal como se muestra a continuación.

```
1 # Definir el método que calcula los números primos
2 def Primos(nMin, nMax):
3     # Crear un ciclo para determinar los números primos
4     Primos_ = np.array([])
5
6     for n in range(nMin, nMax + 1):
7
8         # Crear un ciclo interno para contar los
9         divisores del número n
10        divisores = 0
11        for m in range(1, n+1):
12
13            # Preguntar si m es divisor de n
14            if (n % m == 0):
15                divisores += 1
16
17            # Si divisores es mayor que 2, n no es primo
18            if divisores > 2:
19                break
20
21        # Si la cantidad de divisores de n es dos es un
22        número primo
23        if divisores==2:
24            Primos_ = np.append(Primos_, n)
25
26        # Devolver los números primos
27    return Primos_
28
29 print(Primos(2, 40))
```


El resultado que obtenemos es el siguiente

$$\text{Primos}_{1,\dots,12} = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37]$$

Luego creamos el código para calcular el polinomio interpolador, tal como sigue

```

1 # Definir los nodos
2 nodos_y = Primos(2, 40)
3 print(nodos_y)
4 nodos_x = np.array([n+1 for n in range(len(nodos_y))])
5 print(nodos_x)
6 nodos = (nodos_x, nodos_y)
7
8 # Usar la función de interpolación
9 poly = polyInterpoLagrange(nodos)
10 print(poly)
11
12 # Crear el dominio de graficación y evaluar
13 xMax = len(nodos_x)
14 x = np.linspace(1, xMax, 200)
15 y = evalPoly(poly, x)
16
17 # Graficamos el polinomio y los puntos
18 plt.plot(x, y)
19 plt.scatter(nodos_x, nodos_y, c='red', zorder=3)
20 plt.grid(linestyle='—')
21 plt.xlabel('Números Naturales')
22 plt.ylabel('Números Primos')
23 plt.title('Gráfica del Polinomio Intepolador de Primeros
    '+str(xMax)+' Números Primos')
24 plt.savefig('../Interpolacion-Aproximacion-Polinomial/
    Imagenes/polinomio-numeros-primos.pdf')
25 plt.show()

```

El polinomio interpolador (tomando tres cifras significativas para los coeficientes) esta dado por

$$P_{11}(x) = -2.006E^{-5}x^{11} + 1.45E^{-3}x^{10} - 4.62E^{-2}x^9 + 8.54E^{-1}x^8 \\ - 1.014E^{+1}x^7 + 8.085E^{+1}x^6 - 4.396E^{+2}x^5 + 1.62E^{+3}x^4 \\ - 3.938E^{+3}x^3 + 5.95E^{+3}x^2 - 4.963E^{+3}x + 1.701E^{+3}$$

Como conclusión, observamos que este tipo de experimentos genera números pequeños. Para dimensionar el primer coeficiente del polinomio es

$$-2.00667388E^{-5} = -0.000020067388$$

El gráfico del polinomio interpolador es el que se muestra en la figura siguiente 4.8

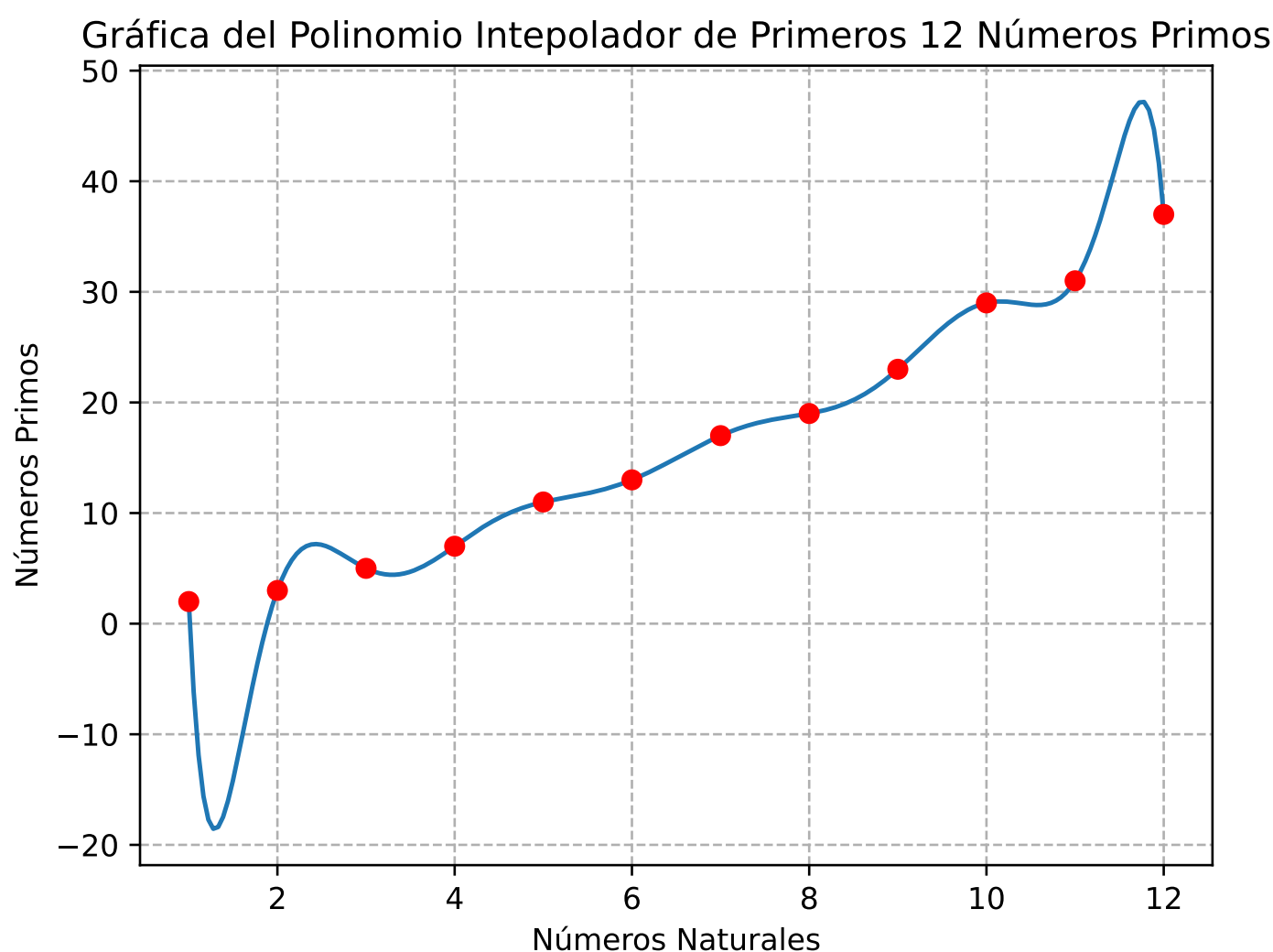


Figura 4.8: Gráfico del Polinomio Interpolador de los Primeros 12 Números Primos

4.5 Método de Neville

Hasta este punto hemos visto dos técnicas para calcular un polinomio interpolador. En el caso de los polinomios interpoladores usando

los polinomios de Lagrange si los cálculos se hicieran manualmente sería muy costoso, por ejemplo calcular en el último ejemplo que realizamos doce (12) polinomios de Lagrange de grado 11, más aun si el experimento exigiera agregar un nodo más a los datos, habría que empezar de nuevo a calcular ahora doce (12) polinomios de Lagrange.

En esta sección vamos a ver cómo ir creando los polinomios interpoladores y dejar con ello un cálculo progresivo de polinomios que nos sirven para calcular un polinomio nuevo agregando datos en el proceso, esto nos permite por ejemplo no perder el trabajo realizado.

Vamos a definir el cálculo de un polinomio interpolador parcial de $n+1$ nodos como

$$P_{0,1,\dots,k}(x)$$

con $k \leq n$, veamos el siguiente ejemplo.

Ejemplo 5:

Calcular los polinomios interpoladores parciales a partir de los siguientes nodos

$$(1, 2), (2, 3), (3, 5), (4, 7), (5, 11)$$

Polinomios de Grado Uno ⁽¹⁾

Para calcular los polinomios parciales de grado uno (1) vamos a usar la siguiente fórmula

$$P_{i,j}(x) = \frac{(x - x_i) P_j - (x - x_j) P_i}{x_j - x_i}$$

donde $j = i + 1$.

Empecemos calculando los polinomios parciales de grado uno (1), teniendo en cuenta que los polinomios de grado cero (0), corresponden a los valores de las ordenadas de los nodos, así

$$P_0 = 2, P_1 = 3, P_2 = 5, P_3 = 7, P_4 = 11, P_5 = 13 \text{ y } P_6 = 17$$

○ Para $P_{0,1}(x)$ con $(1, 2)$ y $(2, 3)$ tenemos

$$\begin{aligned} P_{0,1}(x) &= \frac{(x-1) \cdot 3 - (x-2) \cdot 2}{2-1} \\ &= x+1 \end{aligned}$$

○ Para $P_{1,2}(x)$ con $(2, 3)$ y $(3, 5)$ tenemos

$$\begin{aligned} P_{1,2}(x) &= \frac{(x-2) \cdot 5 - (x-3) \cdot 3}{3-2} \\ &= 2x-1 \end{aligned}$$

○ Para $P_{2,3}(x)$ con $(3, 5)$ y $(4, 7)$ tenemos

$$\begin{aligned} P_{2,3}(x) &= \frac{(x-3) \cdot 7 - (x-4) \cdot 5}{4-3} \\ &= 2x-1 \end{aligned}$$

○ Para $P_{3,4}(x)$ con $(4, 7)$ y $(5, 11)$ tenemos

$$\begin{aligned} P_{3,4}(x) &= \frac{(x-4) \cdot 11 - (x-5) \cdot 7}{5-4} \\ &= 4x-9 \end{aligned}$$

Polinomios de Grado Dos (2)

Para calcular los polinomios parciales de grado dos (2) vamos a usar la siguiente fórmula

$$P_{i,j,k}(x) = \frac{(x-x_i) P_{j,k} - (x-x_k) P_{i,j}}{x_k - x_i}$$

con $j = i+1$ y $k = i+2$

○ Para $P_{0,1,2}(x)$ con $(1, 2)$ y $(3, 5)$ tenemos

$$\begin{aligned} P_{0,1,2}(x) &= \frac{(x-1) P_{1,2} - (x-3) P_{0,1}}{3-1} \\ &= \frac{(x-1)(2x-1) - (x-3)(x+1)}{2} \\ &= \frac{1}{2}(x^2 - x + 4) \end{aligned}$$

○ Para $P_{1,2,3}(x)$ con $(2, 3)$ y $(4, 7)$ tenemos

$$\begin{aligned} P_{1,2,3}(x) &= \frac{(x-2)P_{2,3} - (x-4)P_{1,2}}{4-2} \\ &= \frac{(x-2)(2x-1) - (x-4)(2x-1)}{2} \\ &= 2x-1 \end{aligned}$$

○ Para $P_{2,3,4}(x)$ con $(3, 5)$ y $(5, 11)$ tenemos

$$\begin{aligned} P_{2,3,4}(x) &= \frac{(x-3)P_{3,4} - (x-5)P_{2,3}}{5-3} \\ &= \frac{(x-3)(4x-9) - (x-5)(2x-1)}{2} \\ &= x^2 - 5x + 11 \end{aligned}$$

Polinomio de Grado Tres (3)

Para calcular los polinomios parciales de grado tres (3) vamos a usar la siguiente fórmula

$$P_{i,j,k,l}(x) = \frac{(x-x_i)P_{j,k,l} - (x-x_l)P_{i,j,k}}{x_l - x_i}$$

○ Para $P_{0,1,2,3}(x)$ con $(1, 2)$ y $(4, 7)$ tenemos

$$\begin{aligned} P_{0,1,2,3}(x) &= \frac{(x-1)P_{1,2,3} - (x-4)P_{0,1,2}}{4-1} \\ &= \frac{(x-1)(2x-1) - (x-4)\frac{1}{2}(x^2-x+4)}{3} \\ &= -\frac{x^3}{6} + \frac{3x^2}{2} - \frac{7x}{3} + 3 \end{aligned}$$

○ Para $P_{1,2,3,4}(x)$ con $(2, 3)$ y $(5, 11)$ tenemos

$$\begin{aligned} P_{1,2,3,4}(x) &= \frac{(x-2)P_{2,3,4} - (x-5)P_{1,2,3}}{5-2} \\ &= \frac{(x-2)(x^2-5x+11) - (x-5)(2x-1)}{3} \\ &= \frac{x^3}{3} - 3x^2 + \frac{32x}{3} - 9 \end{aligned}$$

Para polinomio de grado cuatro ⁽⁴⁾

Para calcular los polinomios parciales de grado cuatro ⁽⁴⁾ vamos a usar la siguiente fórmula

$$P_{i,j,k,l,m}(x) = \frac{(x - x_i) P_{j,k,l,m} - (x - x_m) P_{i,j,k,l}}{x_m - x_i}$$

Para $P_{0,1,2,3,4}(x)$ con $(1, 2)$ y $(5, 11)$ tenemos

$$\begin{aligned} P_{0,1,2,3,4}(x) &= \frac{(x - 1) P_{1,2,3,4} - (x - 5) P_{0,1,2,3}}{5 - 1} \\ &= \frac{(x - 1) \left(\frac{x^3}{3} - 3x^2 + \frac{32x}{3} - 9 \right) - (x - 5) \left(-\frac{x^3}{6} + \frac{3x^2}{2} - \frac{7x}{3} + 3 \right)}{4} \\ &= \frac{x^4}{8} - \frac{17x^3}{12} + \frac{47x^2}{8} - \frac{103x}{12} + 6 \end{aligned}$$

Veamos a continuación la gráfica de la función polinómica encontrada y los puntos de interpolación.

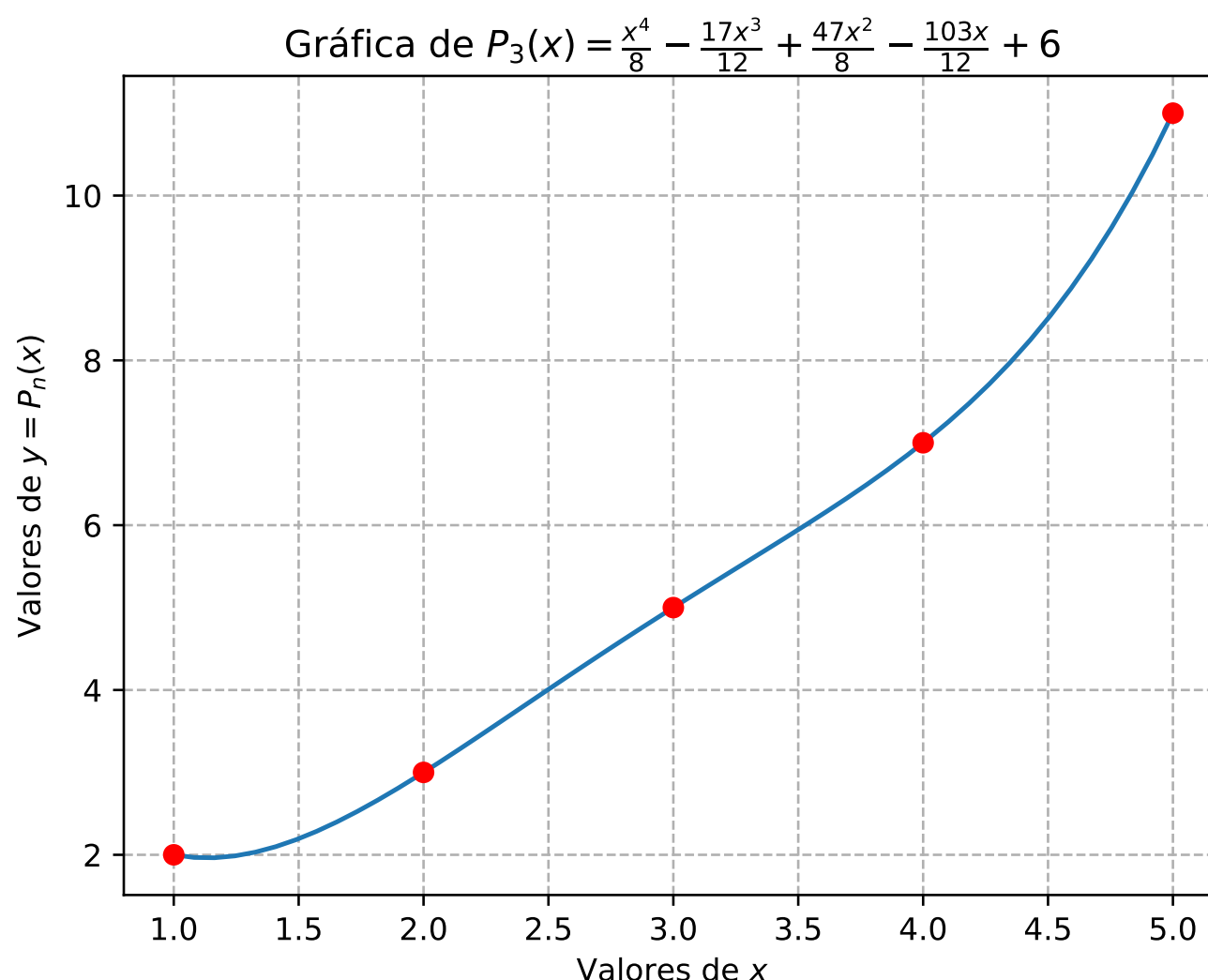


Figura 4.9: Gráfica de la Función Polinómica $f(x) = \frac{x^4}{8} - \frac{17x^3}{12} + \frac{47x^2}{8} - \frac{103x}{12} + 6$

4.6 Diferencias Divididas de Newton

Este método es similar al método de Neville, ya que permite ir agregando nuevos nodos al último polinomio calculado. Veamos un ejemplo particular y luego observamos el método general.

Tenemos los siguientes puntos o nodos a interpolar: $(1, 1)$, $(2, \phi)$, $(3, e)$ y $(4, \pi)$. Este experimento consiste en encontrar un polinomio de grado tres (3) que contenga exactamente a estos números tan famosos de las matemáticas.

○ 1

○ $\phi = \frac{1 + \sqrt{5}}{2}$

○ $e \approx 2.718281828459$

○ $\pi = 3.1415926535898$

Empecemos considerando el polinomio de grado cero

$$P_0(x_0 = 1) = 1$$

luego queremos que cuando evaluemos $P_1(x_1 = 2) = \phi$ pero conservando el valor anterior, entonces podemos definir

$$P_1(x_1) = 1 + a_1(x - x_0)$$

si evaluamos $x = x_0$ de elimina el factor de a_1 y da como resultado y_0 , y si evaluamos $x = x_1$ podemos obtener el valor de y_1 , así para el nodo $(2, \phi)$

$$\phi = 1 + a_1(2 - 1)$$

$$\phi = 1 + a_1$$

$$a_1 = \phi - 1$$

dándonos un polinomio de la forma

$$P_1(x) = 1 + (\phi - 1)(x - 1)$$

si evaluamos $x = 1$ entonces $P_1(1) = 1$ y si evaluamos

$$\begin{aligned} P_1(2) &= 1 + (\phi - 1)(2 - 1) \\ &= 1 + \phi - 1 \\ &= \phi \end{aligned}$$

Es decir que hemos logrado interpolar correctamente a los dos primeros nodos.

Para el nodo $(3, e)$ tenemos

$$P_2(x) = 1 + (\phi - 1)(x - 1) + a_2(x - 1)(x - 2)$$

se le agrega un nuevo polinomio cuadrático, evaluamos en $x = 3$ y despejamos a a_2

$$\begin{aligned} P_2(3) &= 1 + (\phi - 1)(3 - 1) + a_2(3 - 1)(3 - 2) \\ e &= 1 + 2(\phi - 1) + 2a_2 \\ e &= 1 + 2\phi - 2 + 2a_2 \\ e - 2\phi + 1 &= 2a_2 \\ a_2 &= \frac{e - 2\phi + 1}{2} \\ &= \frac{e}{2} - \phi + \frac{1}{2} \end{aligned}$$

así el polinomio de grado dos (2) es

$$P_2(x) = 1 + (\phi - 1)(x - 1) + \left[\frac{e}{2} - \phi + \frac{1}{2} \right] (x - 1)(x - 2)$$

Para que se vea mejor hay que quitar todos los paréntesis, entonces

$$P_2(x) = 1 + \phi x - \phi - x + 1$$

4.6.1 Proceso General

Consideremos un conjunto de puntos $X_0(x_0, y_0), X_1(x_1, y_1), \dots, X_n(x_n, y_n)$ y consideremos cada polinomio linealmente independiente

$$a_0, a_1(x - x_0), a_2(x - x_0)(x - x_1), \dots, a_n(x - x_0) \cdots (x - x_{n-1})$$

entonces el polinomio de grado cero (0) está dado por

$$P_0(x_0) = y_0$$

así $a_0 = y_0$

luego el polinomio de grado uno (1) se sigue como

$$a_0 + a_1(x_1 - x_0) = P_1(x_1)$$

sabemos que

$$P_1(x_1) = y_1 \text{ y } a_0 = y_0$$

entonces

$$\begin{aligned} y_0 + a_1(x - x_0) &= y_1 \\ a_1(x - x_0) &= y_1 - y_0 \\ a_1 &= \frac{y_1 - y_0}{x_1 - x_0} \end{aligned}$$

esto lo podemos ver como una pendiente de la recta, y la ecuación del polinomio se transforma en

$$P_1(x) = y_0 + \left(\frac{y_1 - y_0}{x_1 - x_0} \right) (x - x_0) \quad (4.4)$$

que es justamente la recta que pasa por los dos primeros nodos X_0, X_1 .

Consideremos ahora el nodo $X_2(x_2, y_2)$ tenemos

$$P_2(x_2) = a_0 + a_1(x_2 - x_0) + a_2(x - x_0)(x - x_1)$$

del cual conocemos:

$$\bigcirc a_0 = y_0$$

$$\bigcirc a_1 = \left(\frac{y_1 - y_0}{x_1 - x_0} \right)$$

$$\bigcirc P_2(x_2) = y_2$$

Entonces el polinomio se transforma en

$$y_0 + \left(\frac{y_1 - y_0}{x_1 - x_0} \right) (x_2 - x_0) + a_2 (x_2 - x_0) (x_2 - x_1) = y_2 \quad (4.5)$$

debemos notar que la expresión

$$y_0 + \left(\frac{y_1 - y_0}{x_1 - x_0} \right) (x_2 - x_0)$$

es el polinomio 4.4 evaluado en $x = x_2$ por lo cual la ecuación 4.5 se transforma en

$$\begin{aligned} P_1(x_2) + a_2(x_2 - x_0)(x_2 - x_1) &= y_2 \\ a_2(x_2 - x_0)(x_2 - x_1) &= y_2 - P_1(x_2) \\ a_2 &= \frac{y_2 - P_1(x_2)}{(x_2 - x_0)(x_2 - x_1)} \end{aligned}$$

donde $P_1(x_2)$ es una especie de término anterior “ y_1 ”. Esta anotación se hace para mostrar que el término que se está calculando es una “forma” de pendiente. De aquí es donde se da el nombre de “Diferencias Divididas”.

Del proceso anterior ya tenemos el coeficiente de a_2 por lo cual el polinomio de grado dos (2) tiene la forma

$$P_2(x) = y_0 + \left(\frac{y_1 - y_0}{x_1 - x_0} \right) (x - x_0) + \frac{y_2 - P_1(x_2)}{(x_2 - x_0)(x_2 - x_1)} (x - x_0)(x - x_1)$$

Consideremos un último nodo para generalizar la forma recursiva, $X_3(x_3, y_3)$ entonces

$$P_3(x_3) = a_0 + a_1(x_3 - x_0) + a_2(x_3 - x_0)(x_3 - x_1) + a_3(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)$$

de donde sabemos que $P_3(x_3) = y_3$, además,

$$a_0 + a_1(x_3 - x_0) + a_2(x_3 - x_0)(x_3 - x_1) = P_2(x_3)$$

por lo que el polinomio se transforma en

$$\begin{aligned} P_2(x_3) + a_3(x_3 - x_0)(x_3 - x_1)(x_3 - x_2) &= y_3 \\ a_3(x_3 - x_0)(x_3 - x_1)(x_3 - x_2) &= y_3 - P_2(x_3) \\ a_3 &= \frac{y_3 - P_2(x_3)}{(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)} \end{aligned}$$

así que el término a_n se puede calcular de la siguiente forma

$$a_n = \frac{y_n - P_{n-1}(x_n)}{\prod_{k=0}^{n-1} (x_n - x_k)}$$

Con esto estamos listos para formular un código que nos permita ir calculando de forma recursiva cada uno de los polinomios.

4.7 Nodos Equiespaciados

Ya se ha trabajado con diferentes tipos de interpolación, ahora supongamos que los nodos están equiespaciados, es decir que $x_i - x_{i-1} = h$, siempre tenemos la misma distancia entre los nodos.

Así se puede representar a los nodos como

$$x_i = x_0 + i \cdot h, \quad i = 0, 1, \dots, n$$

Por ejemplo para escribir $x_1 = x_0 + h$, $x_2 = x_0 + 2h$, así hasta $x_n = x_0 + n \cdot h$

Consideremos el conjunto de puntos (nodos)

$$\Omega = \{(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)\}$$

entonces, para

○ Para (x_0, y_0)

$$P_0(x_0) = y_0$$

○ Para (x_1, y_1)

$$P_1(x_1) = P_0(x_0) + a_1(x_1 - x_0)$$

donde $P_1(x_1) = y_1$ y

$$\begin{aligned} x_1 - x_0 &= x_0 + h - x_0 \\ &= h \end{aligned}$$

por lo tanto

$$\begin{aligned}y_1 &= P_0(x_0) + a_1 \cdot h \\y_1 - P_0(x_0) &= a_1 \cdot h \\ \frac{y_1 - P_0(x_0)}{h} &= a_1\end{aligned}$$

de tal manera que

$$P_1(x) = y_0 + \left(\frac{y_1 - P_0(x_0)}{h} \right) (x - x_0)$$

○ Para (x_2, y_2)

$$P_2(x_2) = P_1(x_2) + a_2(x_2 - x_0)(x_2 - x_1)$$

donde:

$$\begin{aligned}\square P_2(x_2) &= y_2 \\ \square x_2 - x_0 &= x_0 + 2h - x_0 = 2h \\ \square x_2 - x_1 &= x_0 + 2h - x_0 - h = h\end{aligned}$$

entonces,

$$\begin{aligned}y_2 &= P_1(x_0 + 2h) + a_2(2h)(h) \\ y_2 - P_1(x_0 + 2h) &= a_2(2 \cdot h^2) \\ \frac{y_2 - P_1(x_0 + 2h)}{2 \cdot h^2} &= a_3\end{aligned}$$

por lo tanto

$$P_2(x) = P_1(x) + \frac{y_2 - P_1(x_0 + 2h)}{2 \cdot h^2} (x - x_0)(x - x_1)$$

○ Para (x_3, y_3)

$$P_3(x_3) = P_2(x_3) + a_3(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)$$

donde:

$$\begin{aligned}\square P_3(x_3) &= y_3 \\ \square x_3 - x_0 &= x_0 + 3h - x_0 = 3h \\ \square x_3 - x_1 &= x_0 + 3h - x_0 - h = 2h \\ \square x_3 - x_2 &= x_0 + 3h - x_0 - 2h = h\end{aligned}$$

entonces,

$$y_3 = P_2(x_0 + 3h) + a_3(3h)(2h)(h)$$

$$y_3 - P_2(x_0 + 3h) = a_3(3!h^3)$$

$$\frac{y_3 - P_2(x_0 + 3h)}{3!h^3} = a_3$$

por lo tanto

$$P_3(x) = P_2(x) + \frac{y_3 - P_2(x_0 + 3h)}{3!h^3}(x - x_0)(x - x_1)(x - x_2)$$

○ Finalmente para comprobar la forma k -ésima,

$$P_4(x_4) = P_3(x_4) + a_4(x_4 - x_0)(x_4 - x_1)(x_4 - x_2)(x_4 - x_3)$$

donde:

- $P_4(x_4) = y_4$
- $x_4 - x_0 = x_0 + 4h - x_0 = 4h$
- $x_4 - x_1 = x_0 + 4h - x_0 - h = 3h$
- $x_4 - x_2 = x_0 + 4h - x_0 - 2h = 2h$
- $x_4 - x_3 = x_0 + 4h - x_0 - 3h = h$

entonces,

$$y_4 = P_3(x_0 + 4h) + a_4(4h)(3h)(2h)(h)$$

$$y_4 - P_3(x_0 + 4h) = a_4(4! \cdot h^4)$$

$$\frac{y_4 - P_3(x_0 + 4h)}{4! \cdot h^4} = a_4$$

por lo tanto

$$P_4(x) = P_3(x) + \frac{y_4 - P_3(x_0 + 4h)}{4! \cdot h^4}(x - x_0)(x - x_1)(x - x_2)(x - x_3)$$

De forma general podemos encontrar que la forma general de cualquier a_k es

$$a_k = \frac{y_k - P_{k-1}(x_0 + k \cdot h)}{k! \cdot h^k}$$

4.7.1 Definición de una Función en Python

A continuación se define una función en Python que permite encontrar el polinomio interpolador usando la técnica de los nodos equiespaciados.

```
1 # Definir el método de las diferencias divididas
2 def NodosEquiespaciados(nodos, x_0, h, console=False):
3
4     """
5     ##### ***Función:*** NodosEquiespaciados
6     – **Descripción:** Esta función calcula un polinomio
7       de grado  $n$  dados un conjunto de nodos
8     – **Parámetros:**
9       – *nodos:* Es una dupla de la forma  $(x[\dots], y[\dots])$ 
10      – *x_0:* Valor inicial en x
11      – *h:* Espacio o distancia entre los nodos  $x_i$  y
12         $x_{i-1}$ 
13      – *console:* Valor de tipo booleano para mostrar
14        mensajes por consola
15     – **Valor de Retorno:** Devuelve un diccionario con
16       todos los polinomios  $P_0, \dots, P_n$ 
17     """
18
19     # Obtener los nodos Y
20     y = nodos
21
22     # Definir un diccionario para los polinomios y crear
23     el primer polinomio
24     Poly = { "P0": [y[0]]}
25
26     # Crear un ciclo para recorrer todos los nodos
27     cantNodos = len(y)
28     for n in range(1, cantNodos):
29
30         # Obtener el polinomio anterior
31         if console:
32             print("Iteración ", n, "
33                     _____")
34
35         P_ = Poly[ "P"+str(n-1)]
36         x_n = np.array([x_0 + n*h])
```

```

31     if console:
32         print( "P" + str(n-1), P_)
33         print( "xn: ", x_n)
34
35     # Evaluar el polinomio  $P_{\{n-1\}}$  en  $x_n$ 
36     y_0 = evalPoly(P_, x_n)
37
38     # Calcular la productoria
39     Factorial = 1
40     raices = np.array([])
41     for k in range(n):
42         Factorial = Factorial * (k+1)
43         raices = np.append(raices, x_0 + k*h)
44
45     if console:
46         print( "Raíces: ", raices)
47         print( "Factorial: ", Factorial)
48
49     # Definir el polinomio  $n-1$ 
50     ms_ = np.insert(P_, 0, 0)
51     a_n = (y[n] - y_0) / (Factorial * h**(k+1))
52     ms = a_n * MultiplicacionSintetica(raices)
53
54     if console:
55         print( "an: ", a_n)
56
57     # Sumar los dos polinomios
58     P_n = ms_ + ms
59
60     if console:
61         print( "Pn: ", P_n)
62
63     # Agregar el polinomio encontrado al diccionario
64     Poly[ "P" + str(n) ] = P_n
65
66     # Devolver el polinomio
67     return Poly

```

4.7.2 Ejemplo

En el siguiente archivo se muestra un ejemplo del uso de este método y su respectivo resultado.

Archivo de Programa

El archivo de programa se encuentra en la ruta: *Unidad-004-Interpolacion-Aproximacion-Polinomial*▷ *Programas*▷ *Nodos-Equiespaciados.ipynb*

```
1 # Define una función que genera valores equiespaciados
2 def funcion(x):
3     return np.sin(x)
4
5 # Crear los nodos equiespaciados
6 x = np.linspace(0, 2*np.pi, 24)
7
8 # Calcular el valor del h
9 h = x[1]-x[0]
10
11 # Calcular los valores de y
12 y_nodos = funcion(x)
13
14 # Usar el método NodosEquiespaciados para calcular el
    polinomio interpolador
15 poly = NodosEquiespaciados(y_nodos, x[0], h)
16 sPoly, pPoly = poly.popitem()
17
18 # Mostrar los coeficientes del polinomio interpolador
19 print(pPoly)
20
21 # Graficar el polinomio interpolador y los nodos
22 y_Poly = evalPoly(pPoly, x)
23 plt.plot(x, y_Poly)
24 plt.scatter(x, y_nodos, c='red', zorder=3)
25 plt.grid(linestyle='—')
26 plt.title('Gráfica del Polinomio Interpolador y los
    Nodos')
```



```

27 plt.xlabel('Valores de $x$')
28 plt.ylabel('Valores de $y$')
29
30 # Guardar y mostrar la gráfica
31 plt.savefig(' ../ Interpolacion-Aproximacion-Polinomial/
    Imagenes/Ejemplo-Nodos-Equiespaciados .pdf')
32 plt.show()

```

Al graficar el polinomio interpolador y los nodos obtenemos el siguiente resultado

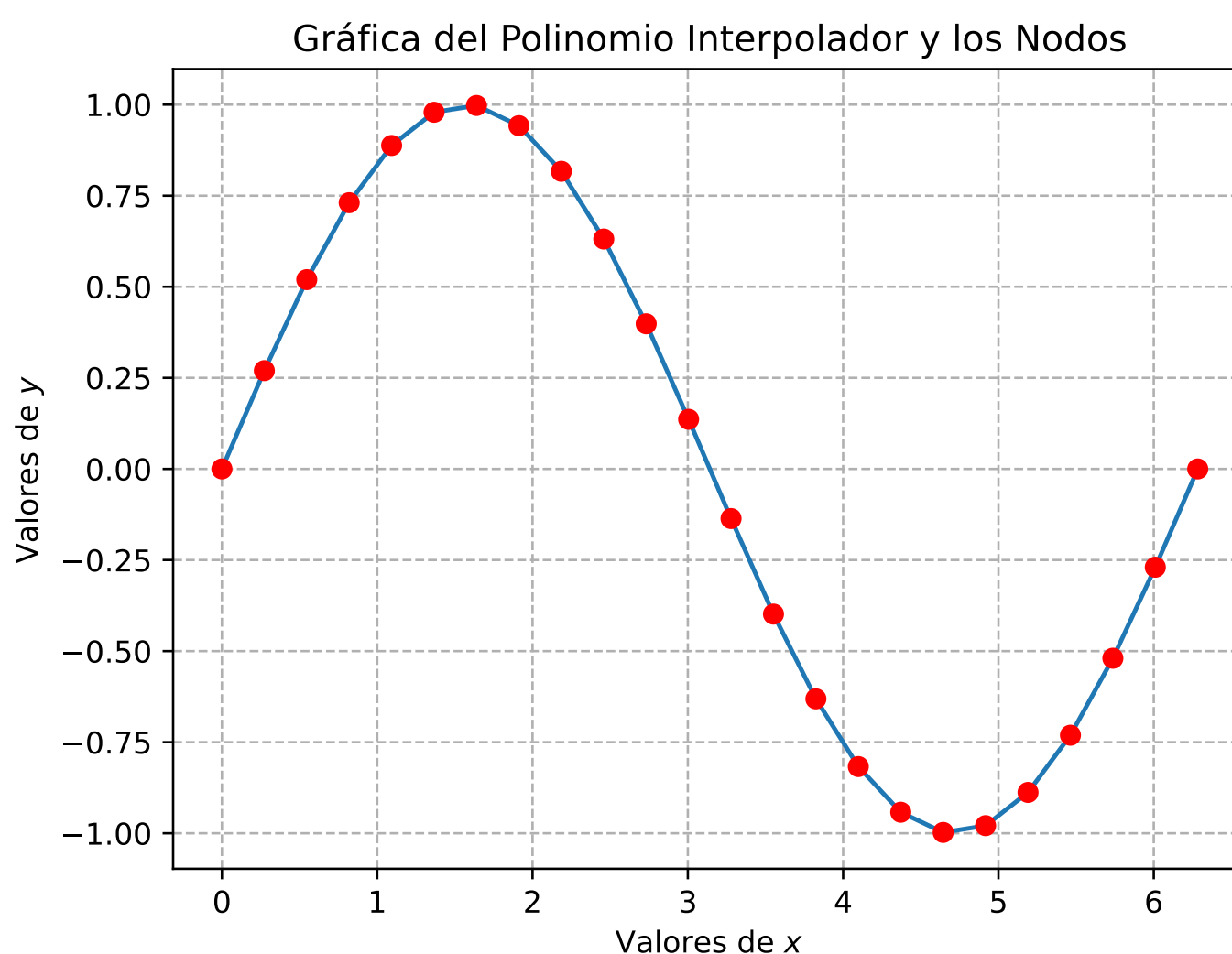


Figura 4.10: Gráfica del Polinomio Interpolador y sus Nodos

4.8 Interpolación de Hermite

Hasta este punto se ha trabajado con varias técnicas para calcular el polinomio interpolador de un conjunto de nodos Ω . Sin embargo no se puede garantizar que la derivada de ese polinomio interpolador en un nodo x_i determinado sea el mismo que el de la función generadora $f'(x_i)$.

Al polinomio que cumple estas características se le llama polinomio **Osculador** de aproximación de f en los nodos x_0, x_1, \dots, x_n .

4.8.1 Polinomios de Hermite

Un caso particular del caso anterior se resuelve mediante el método de Hermite.

¿Qué queremos resolver?

Tenemos x_0, x_1, \dots, x_n , $n + 1$ nodos distintos. Sea $f \in \mathcal{C}^1$ una función de clase \mathcal{C}^1 , esto quiere decir que la función y su derivada son continuas, así, queremos hallar un polinomio $P(x)$ de tal forma que la función f y el polinomio anterior coincidan en los nodos x_0, x_1, \dots, x_n es decir, que tengan la misma recta tangente, así,

$$\begin{aligned} P(x_i) &= f(x_i) \text{ y} \\ P'(x_i) &= f'(x_i), \text{ para } i = 0, 1, \dots, n \end{aligned}$$

Cuando calculamos el polinomio interpolador con Lagrange, teníamos que

$$P_n(x) = \sum_{k=0}^n y_k L_{n,k}(x)$$

es decir

$$y_0 \cdot L_{n,0}(x) + y_1 \cdot L_{n,1}(x) + \dots + y_n \cdot L_{n,n}(x)$$

esto recordando que cada polinomio de Lagrange tiene una característica, es 1 si es evaluado en el nodo sobre el cual fue calculado, por ejemplo $L_{n,0}(x)$ es 1 si se evalúa en $x = x_0$, mientras que $L_{n,k}(x)$ es 1 si se evalúa en $k = x_k$ esto permite entender que al multiplicar por cada y_k se obtiene el polinomio interpolador que pasa por los puntos deseados.

El problema que se tiene ahora es que queremos que al calcular la derivada del polinomio interpolador en cada uno de los puntos interpoladores, la derivada sea la misma que la de la función que se quiere interpolar. Para ello debemos asegurarnos que la expresión

$$f_k(x) = y_k \cdot L_{n,k}(x_k)$$

sea y_k para $f_k(x_k)$, y se anule en $f'_k(x_k)$, ¿Cómo podemos modificar la expresión para que ello se cumpla?

Una forma de lograr que se anulen la derivada cuando $x = x_k$ en $f_k(x)$ es realizar manipulaciones algebraicas.

Podemos proponer la siguiente expresión

$$f_k(x) = y_k \cdot L_{n,k}^2(x)$$

es una expresión que al evaluarla en $x = x_k$ sigue siendo y_k y al derivar tenemos,

$$f'_k(x) = 2 \cdot y_k \cdot L'_{n,k}(x) \cdot L_{n,k}(x)$$

que al evaluar en $x = x_k$ da como resultado

$$f'_k(x_k) = 2 \cdot y_k \cdot L'_{n,k}(x_k)$$

esta expresión debería estar multiplicada por $(x - x_k)$ para que al evaluar en $x = x_k$ se anule.

Una nueva proposición para $f_k(x)$ es

$$f_k(x) = y_k \cdot (x - x_k) \cdot L_{n,k}^2(x) \quad (4.6)$$

entonces la derivada esta dada por

$$f'_k(x) = y_k L_{n,k}^2(x) + 2y_k (x - x_k) \cdot L'_{n,k}(x) \cdot L_{n,k}(x)$$

que al evaluar en $x = x_k$ la primera expresión es y_k ya que $L_{n,k}^2(x)$ evaluada en $x = x_k$ es igual a 1 y la segunda expresión se anula por $(x - x_k)$ cuando $x = x_k$.

Aun nos falta agregar una nueva expresión que anule totalmente. Se propone una nueva expresión

$$f_k(x) = y_k \cdot [1 - 2(x - x_k) L'_{n,k}(x)] \cdot L_{n,k}^2(x)$$

al derivar tenemos

$$f'_k(x) = y_k [-2L'_{n,k}(x) - 2(x - x_k) L''_{n,k}(x)] L_{n,k}^2(x) + 2y_k [1 - 2(x - x_k) L'_{n,k}(x)] L'_{n,k}(x)$$

si evaluamos la derivada en $x = x_k$ tenemos que

$$\begin{aligned} f'_k(x_k) &= y_k [-2L'_{n,k}(x_k)] + 2y_k [1] L'_{n,k}(x_k) \\ &= -2y_k L'_{n,k}(x_k) + 2y_k L'_{n,k}(x_k) \\ &= 0 \end{aligned}$$

tal como queríamos encontrar para los polinomios que interpolan a los nodos $(x_i, f(x_i))$.

Ahora necesitamos una expresión que interpole a los puntos cuando se da la derivada, es decir $(x_i, f'(x_i))$, para ello en el proceso anterior encontramos la forma 4.6, que es

$$f_k(x) = y_k \cdot (x - x_k) \cdot L_{n,k}^2(x)$$

para diferenciar del caso anterior vamos a considerar el nodo (x_i, \hat{y}_k) , así tenemos

$$\hat{f}_k(x) = \hat{y}_k \cdot (x - x_k) \cdot L_{n,k}^2(x)$$

al derivar y evaluar en $x = x_k$, tenemos

$$\begin{aligned} \hat{f}'_k(x_k) &= \hat{y}_k \cdot L_{n,k}^2(x_k) + 2\hat{y}_k \cdot (x - x_k) \cdot L_{n,k}(x) \\ &= \hat{y}_k \cdot L_{n,k}^2(x_k) \\ &= \hat{y}_k \end{aligned}$$

tal como queríamos obtener.

Con estas expresiones podemos proponer una nueva forma de interpolar un polinomio que cumpla las condiciones

$$P(x_i) = f(x_i) \text{ y } P'(x_i) = f'(x_i)$$

así tenemos

$$P(x) = \sum_{k=0}^n y_k \cdot H_{n,k}(x) + \sum_{k=0}^n \hat{y}_k \cdot \hat{H}_{n,k}(x)$$

donde los polinomios $H_{n,k}(x)$ y $\hat{H}_{n,k}(x)$ son respectivamente

$$\begin{aligned} H_{n,k}(x) &= (1 - 2(x - x_k) \cdot L'_{n,k}(x_k)) L_{n,k}^2(x) \\ \hat{H}_{n,k}(x) &= (x - x_k) L_{n,k}^2(x) \end{aligned}$$

Ejemplo 6:

Calcular el polinomio interpolador de Hermite en los nodos $x_0 = 0$, $x_1 = 1$, $x_2 = 3$ y $x_3 = 5$ para la función

$$f(x) = x \cdot \sin\left(\frac{\pi}{2}x\right)$$

Para encontrar los valores de y evaluamos la función en los nodos establecidos y para encontrar los valores de y' debemos derivar la

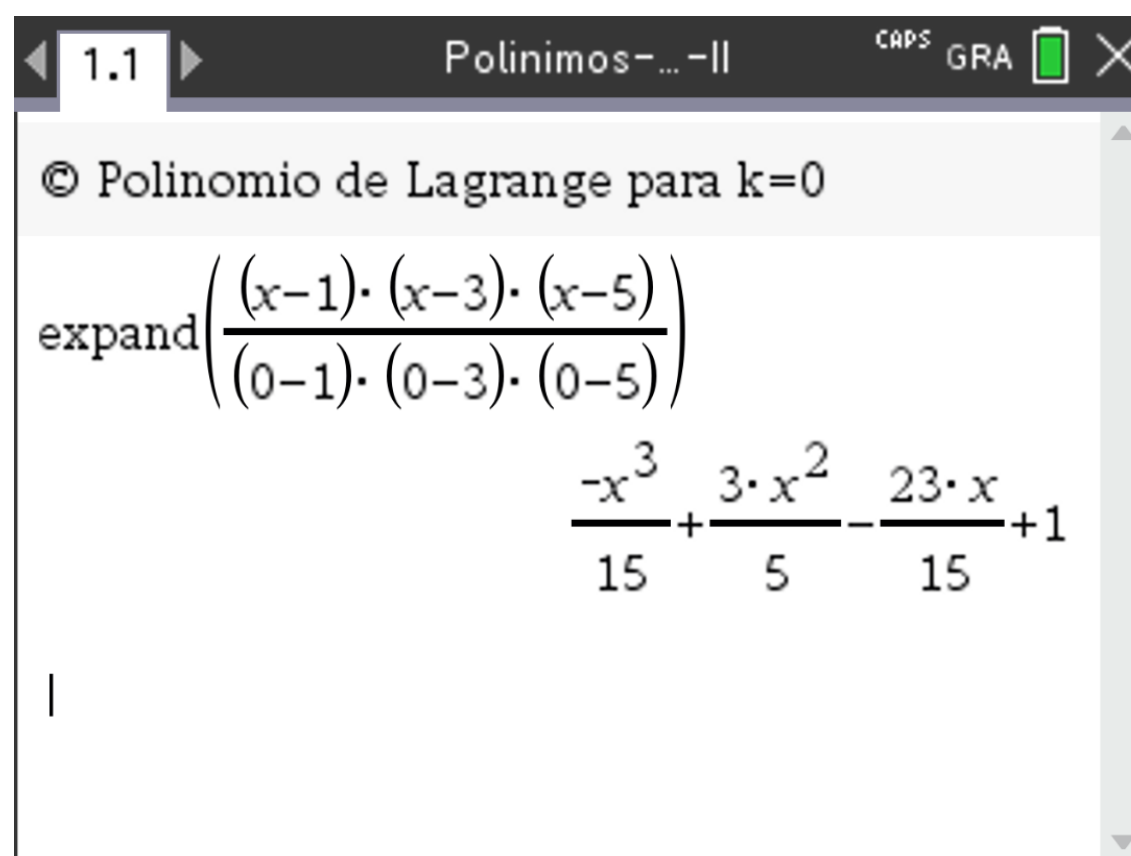
función, así,

$$\begin{aligned} f'(x) &= \sin\left(\frac{\pi}{2}x\right) + x \cdot \frac{\pi}{2} \cdot \cos\left(\frac{\pi}{2}x\right) \\ &= \sin\left(\frac{\pi}{2}x\right) + \frac{\pi x}{2} \cos\left(\frac{\pi}{2}x\right) \end{aligned}$$

así tenemos el siguiente conjunto de nodos

x	0	1	3	5
y	0	1	-3	5
y'	0	1	-1	1

para calcular los polinomios H_3 y \hat{H}_3 debemos calcular cada uno de los polinomios de Lagrange, entonces tenemos los siguientes cálculos

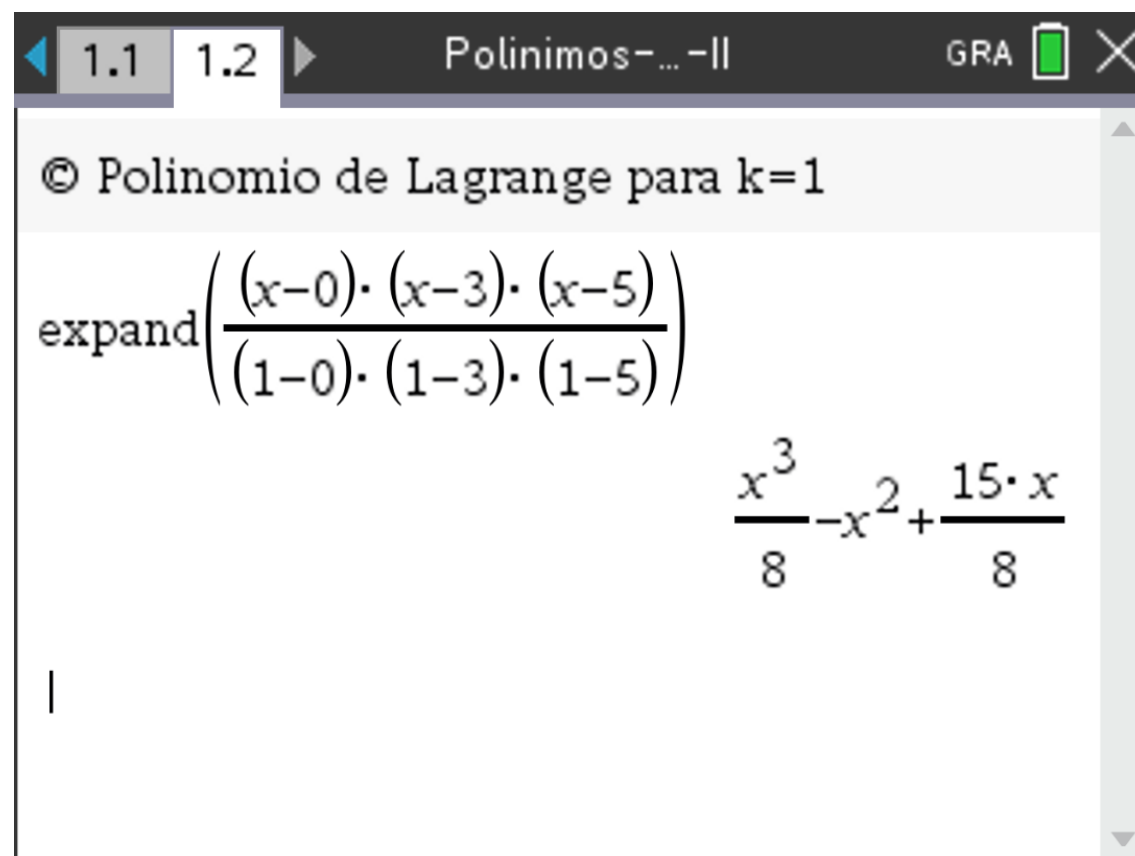


© Polinomio de Lagrange para $k=0$

$$\text{expand}\left(\frac{(x-1) \cdot (x-3) \cdot (x-5)}{(0-1) \cdot (0-3) \cdot (0-5)}\right)$$

$$\frac{-x^3}{15} + \frac{3 \cdot x^2}{5} - \frac{23 \cdot x}{15} + 1$$

Figura 4.11: Cálculo del Polinomio de Lagrange para $k = 0$

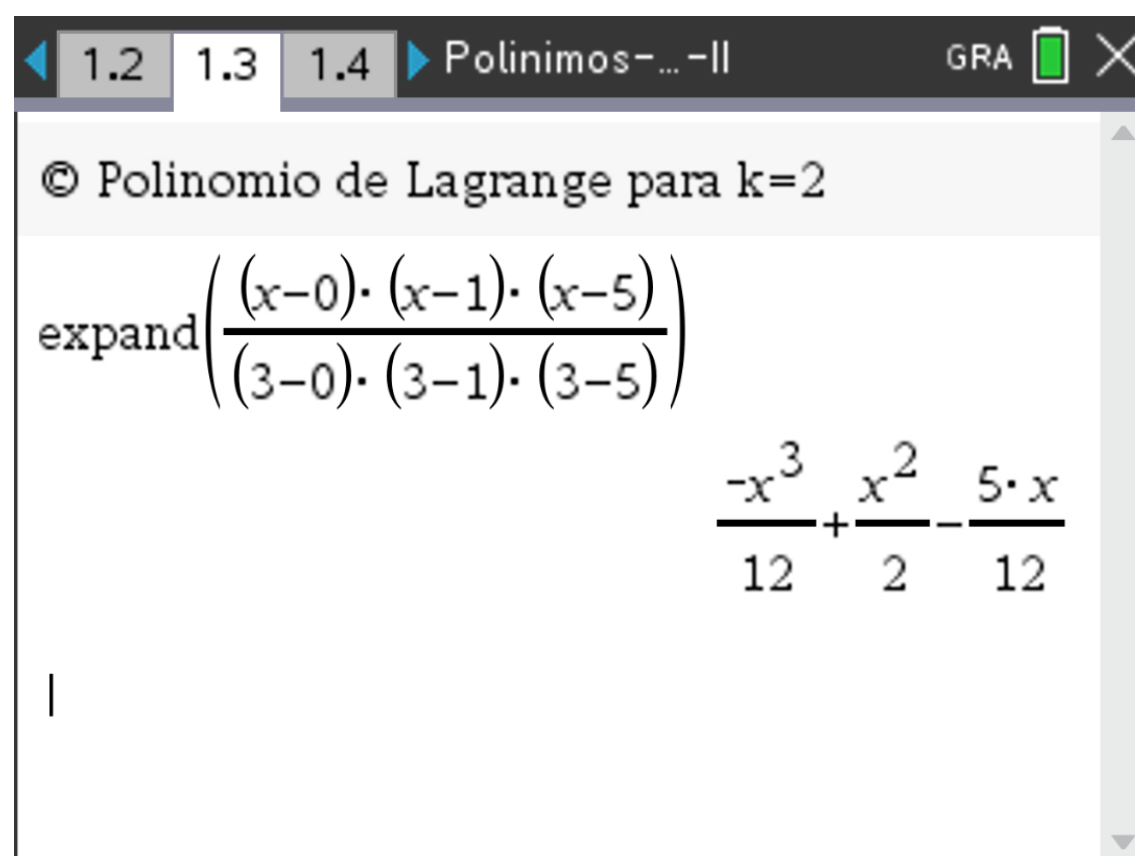


© Polinomio de Lagrange para $k=1$

$$\text{expand}\left(\frac{(x-0) \cdot (x-3) \cdot (x-5)}{(1-0) \cdot (1-3) \cdot (1-5)}\right)$$

$$\frac{x^3}{8} - x^2 + \frac{15 \cdot x}{8}$$

Figura 4.12: Cálculo del Polinomio de Lagrange para $k = 1$



© Polinomio de Lagrange para $k=2$

$$\text{expand}\left(\frac{(x-0) \cdot (x-1) \cdot (x-5)}{(3-0) \cdot (3-1) \cdot (3-5)}\right)$$

$$\frac{-x^3}{12} + \frac{x^2}{2} - \frac{5 \cdot x}{12}$$

Figura 4.13: Cálculo del Polinomio de Lagrange para $k = 2$

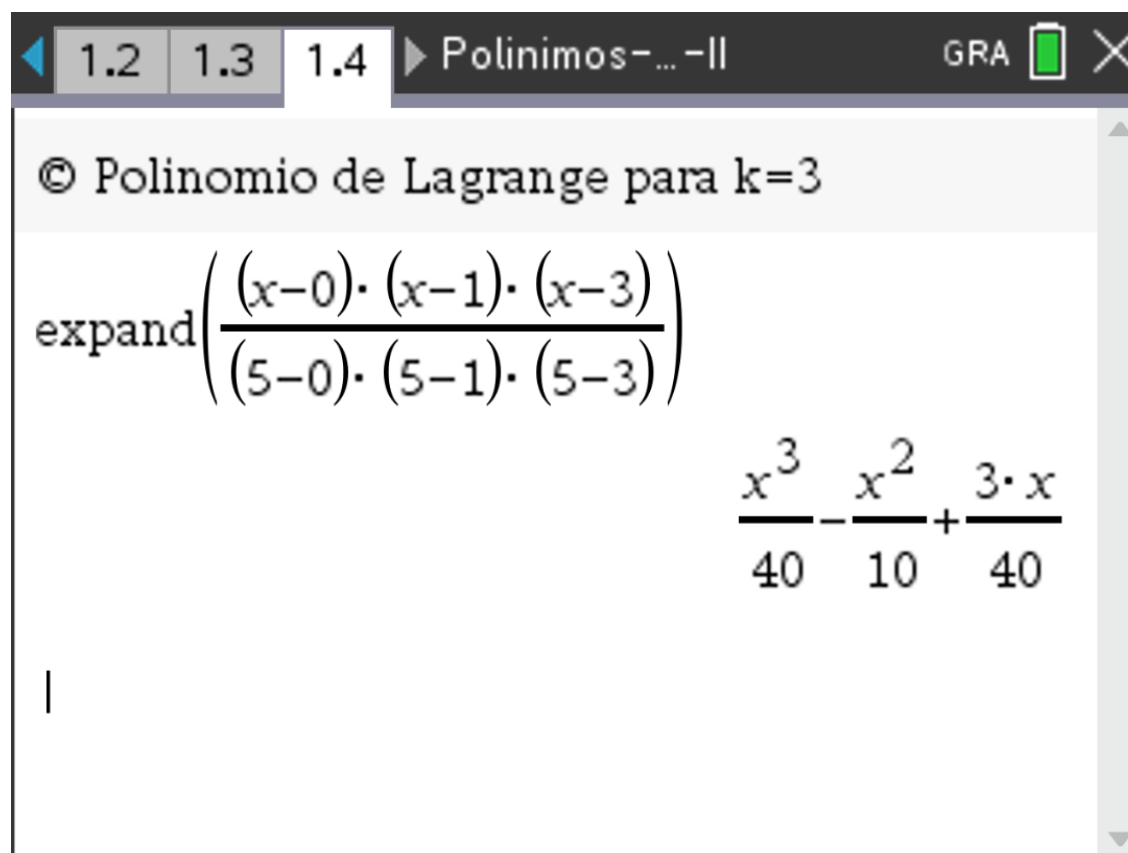


Figura 4.14: Cálculo del Polinomio de Lagrange para $k = 3$

Con estos polinomios encontramos los polinomios $H_{3,k}(x)$ y $\hat{H}_{3,k}(x)$

○ Tenemos el nodo $(0, 0)$ con $L_{3,0}(x) = -\frac{x^3}{15} + \frac{3x^2}{5} - \frac{23x}{15} + 1$ y $L'_{3,0}(x) = -\frac{x^2}{5} + \frac{6x}{5} - \frac{23}{15}$, así,

$$\begin{aligned} H_{3,0}(x) &= [1 - 2(x - 0)L'_{3,0}(0)] \left(-\frac{x^3}{15} + \frac{3x^2}{5} - \frac{23x}{15} + 1 \right)^2 \\ &= \left[1 - 2x \left(-\frac{23}{15} \right) \right] \left(-\frac{x^3}{15} + \frac{3x^2}{5} - \frac{23x}{15} + 1 \right)^2 \\ &= \frac{46}{3375}x^7 - \frac{271}{1125}x^6 + \frac{5572}{3375}x^5 - \frac{6173}{1125}x^4 + \frac{30094}{3375}x^3 - \frac{439}{75}x^2 + 1 \\ &\approx 0.014x^7 - 0.241x^6 + 1.651x^4 - 5.487x^4 + 8.917x^3 - 5.853x^2 + 1.000 \end{aligned}$$

luego

$$\begin{aligned} \hat{H}_{3,0}(x) &= (x - 0) \left(-\frac{x^3}{15} + \frac{3x^2}{5} - \frac{23x}{15} + 1 \right)^2 \\ &= \frac{1}{225}x^7 - \frac{2}{225}x^6 + \frac{127}{225}x^5 - \frac{148}{75}x^4 + \frac{799}{225}x^3 - \frac{46}{15}x^2 + x \\ &\approx 0.004x^7 - 0.080x^6 + 0.567x^5 - 1.973x^4 + 3.551x^3 - 3.067x^2 + x \end{aligned}$$

○ Tenemos el nodo $(1, 1)$ con $L_{3,1}(x) = \frac{1}{8}x^3 - x^2 + \frac{15}{8}x$ y $L'_{3,1}(x) =$

$$\frac{3}{8}x^2 - 2x + \frac{15}{8}, \text{ así}$$

$$\begin{aligned} H_{3,1}(x) &= \left[1 - 2(x-1) \left(\frac{1}{4} \right) \right] \left(\frac{1}{8}x^3 - x^2 + \frac{15}{8} \right)^2 \\ &= -\frac{1}{128}x^7 + \frac{19}{128}x^6 - \frac{71}{64}x^5 + \frac{261}{64}x^4 - \frac{945}{128}x^3 + \frac{675}{128}x^2 \\ &\approx -0.008x^7 + 0.148x^6 - 1.109x^5 + 4.078x^4 - 7.383x^3 + 5.273x^2 \end{aligned}$$

luego

$$\begin{aligned} \hat{H}_{3,1}(x) &= (x-1) \left(\frac{1}{8}x^3 - x^2 + \frac{15}{8} \right)^2 \\ &= \frac{1}{64}x^7 - \frac{17}{64}x^6 + \frac{55}{32}x^5 - \frac{167}{55}x^4 + \frac{465}{64}x^3 - \frac{225}{65}x^2 \\ &\approx 0.016x^7 - 0.266x^6 + 1.719x^5 - 5.219x^4 + 7.266x^3 - 3.516x^2 \end{aligned}$$

○ Tenemos el nodo $(3, -3)$ con $L_{3,2}(x) = -\frac{1}{12}x^3 + \frac{1}{2}x^2 - \frac{5}{12}x$ y $L'_{3,2}(x) = -\frac{1}{4}x^2 + x - \frac{5}{12}$, así

$$\begin{aligned} H_{3,2}(x) &= \left[1 - 2(x-3) \left(\frac{1}{3} \right) \right] \left(-\frac{1}{12}x^3 + \frac{1}{2}x^2 - \frac{5}{12}x \right)^2 \\ &= -\frac{1}{216}x^7 + \frac{11}{144}x^6 - \frac{25}{54}x^5 + \frac{89}{54}x^4 - \frac{295}{216}x^3 + \frac{25}{48}x^2 \\ &\approx -0.005x^7 + 0.076x^6 - 0.463x^5 + 1.236x^4 - 1.366x^3 + 0.521x^2 \end{aligned}$$

luego

$$\begin{aligned} \hat{H}_{3,2}(x) &= (x-3) \left(-\frac{1}{12}x^3 + \frac{1}{2}x^2 - \frac{5}{12}x \right)^2 \\ &= \frac{1}{144}x^7 - \frac{5}{48}x^6 + \frac{41}{72}x^5 - \frac{11}{8}x^4 + \frac{205}{144}x^3 - \frac{25}{48}x^2 \\ &\approx 0.007x^7 - 0.104x^6 + 0.569x^5 - 1.375x^4 + 1.424x^3 - 0.521x^2 \end{aligned}$$

○ Tenemos el nodo $(5, 5)$ con $L_{3,3}(x) = \frac{1}{40}x^3 - \frac{1}{10}x^2 + \frac{3}{40}x$ y $L'_{3,3}(x) =$

$$\frac{3}{40}x^2 - \frac{1}{5}x + \frac{3}{40}, \text{ así}$$

$$\begin{aligned} H_{3,3}(x) &= \left[1 - 2(x-5) \left(\frac{19}{20} \right) \right] \left(\frac{1}{40}x^3 - \frac{1}{10}x^2 + \frac{3}{40}x \right)^2 \\ &= -\frac{19}{16000}x^7 + \frac{257}{16000}x^6 - \frac{629}{8000}x^5 + \frac{1383}{8000}x^4 - \frac{2691}{16000}x^3 + \frac{189}{3200}x^2 \\ &\approx -0.001x^7 + 0.016x^6 - 0.079x^5 + 0.173x^4 - 0.168x^3 + 0.059x^2 \end{aligned}$$

luego

$$\begin{aligned} \hat{H}_{3,3}(x) &= (x-5) \left(\frac{1}{40}x^3 - \frac{1}{10}x^2 + \frac{3}{40}x \right)^2 \\ &= \frac{1}{1600}x^7 - \frac{13}{1600}x^6 + \frac{31}{800}x^5 - \frac{67}{800}x^4 + \frac{129}{1600}x^3 - \frac{9}{320}x^2 \\ &\approx 6.25 \times 10^{-4}x^7 - 0.008x^6 + 0.039x^5 - 0.084x^4 + 0.081x^3 - 0.028x^2 \end{aligned}$$

El polinomio interpolador tiene la forma

$$\begin{aligned} P_7(x) &= y_0 H_{3,0}(x) + y'_0 \hat{H}_{3,0}(x) + y_1 H_{3,1}(x) + y'_1 \hat{H}_{3,1}(x) \\ &\quad + y_2 H_{3,2}(x) + y'_2 \hat{H}_{3,2}(x) + y_3 H_{3,3}(x) + y'_3 \hat{H}_{3,3}(x) \end{aligned}$$

lo cual nos da como resultado

$$\begin{aligned} P_7(x) &= \frac{17}{1800}x^7 - \frac{17}{100}x^6 + \frac{967}{900}x^5 - \frac{202}{75}x^4 + \frac{3233}{1800}x^3 + \frac{59}{60}x^2 \\ &\approx 0.009x^7 - 0.170x^6 + 1.074x^5 - 2.693x^4 + 1.796x^3 + 0.983x^2 \end{aligned}$$

4.8.2 Código de Python

A continuación se presenta el código realizado en Python para calcular los polinomios de Hermite. Se debe tener en cuenta que se han usado los métodos construidos anteriormente.

```

1 # Definir el método de interpolación de Hermite
2 def Hermite(nodos, console=False):
3
4     """
5     ##### ***Función*** Hermite
6     — **Descripción:** Permite calcular el polinomio
      interpolador de un conjunto de nodos (x[...], y
      [...], y' [...]) a través del método de los
      polinomios de Hermite
    """

```

```
7  — **Parámetros:**
8      — *nodos: Tupla de los nodos que se quieren
          interpolar (x[...], y[...], y' [...])
9      — *console: Valor boolean que permite mostrar
          el proceso de cálculo si esta en True
10 — **Valor de Retorno:** Un vector con los
          coeficientes del polinomio interpolador
11 "" ""
12
13 # Obtener los nodos x,y y dy
14 nodos_x, nodos_y, nodos_dy = nodos
15
16 # Calcular la cantidad de nodos a trabajar y el
          grado del polinomio base
17 cantNodos = len(nodos_x)
18 grado = cantNodos - 1
19
20 # Definir el resultado del polinomio interpolador 2*
          grado + 1
21 PolyInterp = np.array([0*i for i in range(2*grado +
          2)])
22
23 # Crear un ciclo para calcular cada uno de los
          polinomios de Lagrange, H y \hat(H)
24 for k in range(cantNodos):
25
26     # Calcular el polinomio de lagrange en k
27     if console:
28         print( " ")
29         print( "—— Para el nodo k =",k, "
          _____")
30
31     L = polyLagrange(nodos_x, k)
32     L2 = polyLagrangeCuadrado(nodos_x, k)
33
34     if console:
35         print( "L(x) =",L)
```

```

36         print( "L2(x) =", L2)
37
38     # Calcular la derivada del polinomio de Lagrange
39     # y su evaluación en el nodo x_k
40     LDer_K = DerivadaPolinomio(L)
41     if console:
42         print( "Dx(L) =", LDer_K)
43
44     xk = np.array([nodos_x[k]])
45     vLDer_K = evalPoly(LDer_K, xk)[0]
46
47     if console:
48         print( "x_" + str(k) + " =", nodos_x[k], "; Dx(x_k) =", vLDer_K)
49
50     # Usar multiplicación sintética para multiplicar
51     # L^2 por [1-2(x-x_k)L'(x)] y Hallar H(x)
52     r = -1/(2*vLDer_K) - nodos_x[k]
53     H = r*L2
54
55     H = np.insert(H, 0, 0)
56     H = np.append(L2, 0) + H
57
58     # Se debe multiplicar el resultado por -2*
59     # vLDer_K por la factorización realizada en la
60     # Multiplicación Sintética
61     H = -2*vLDer_K*H
62
63     if console:
64         print( "H: ", H)
65
66     # Usar multiplicación sintética para multiplicar
67     # L^2 por (x-x_k) y Hallar \hat{H}(x)
68     r = -nodos_x[k]
69     H_ = r*L2
70     H_ = np.insert(H_, 0, 0)
71     H_ = np.append(L2, 0) + H_

```

```
67         if console:
68             print( "H_: ", H_)
69
70
71         # Crear la suma del polinomio interpolador  $y_k \cdot H(x) + dy_k \cdot \hat{H}(x)$ 
72         PolyInterp = PolyInterp + nodos_y[k]*H +
73             nodos_dy[k]*H_
74
75     # Devolver el resultado del polinomio interpolador
76     return PolyInterp
```

4.8.2.1 Uso del Método

A continuación se muestra el uso de éste método con el ejemplo que se desarrolló anteriormente.

```
1 # Definir los nodos de x, y y la derivada
2 nodos_x = np.array([0, 1, 3, 5])
3 nodos_y = np.array([0, 1, -3, 5])
4 nodos_dy = np.array([0, 1, -1, 1])
5
6 # Crear la Tupla para usar los polinomios de Hermite
7 nodos = (nodos_x, nodos_y, nodos_dy)
8 poly = Hermite(nodos)
9 print( "Polinomio Interpolador: ", poly)
10
11 # Crear un dominio de graficación, evaluar la función y
12   el polinomio calculado
13 x = np.linspace(0, 5, 100)
14 P = evalPoly(poly, x)
15 y = funcion(x)
16
17 # Realizar la gráfica de la función
18 plt.plot(x, y)
19 plt.plot(x, P)
20 plt.scatter(nodos_x, nodos_y, c='red', zorder=4)
```

```

21 # Configurar la gráfica
22 plt.title('Gráfica del Polinomio Interpolador')
23 plt.xlabel('Valores de $x$')
24 plt.ylabel('Valores de $y$')
25 plt.grid(linestyle='—')
26 plt.legend(['$y=f(x)$', '$y=P(x)$'])
27 plt.show()

```

Como resultado tenemos lo siguiente:

```

1 Polinomio Interpolador: [ 0.00944444 -0.17
    1.07444444 -2.69333333  1.79611111  0.98333333
2    0.          0.          ]

```

y la gráfica de la función, el polinomio interpolador y los puntos de interpolación se muestran en la siguiente figura.

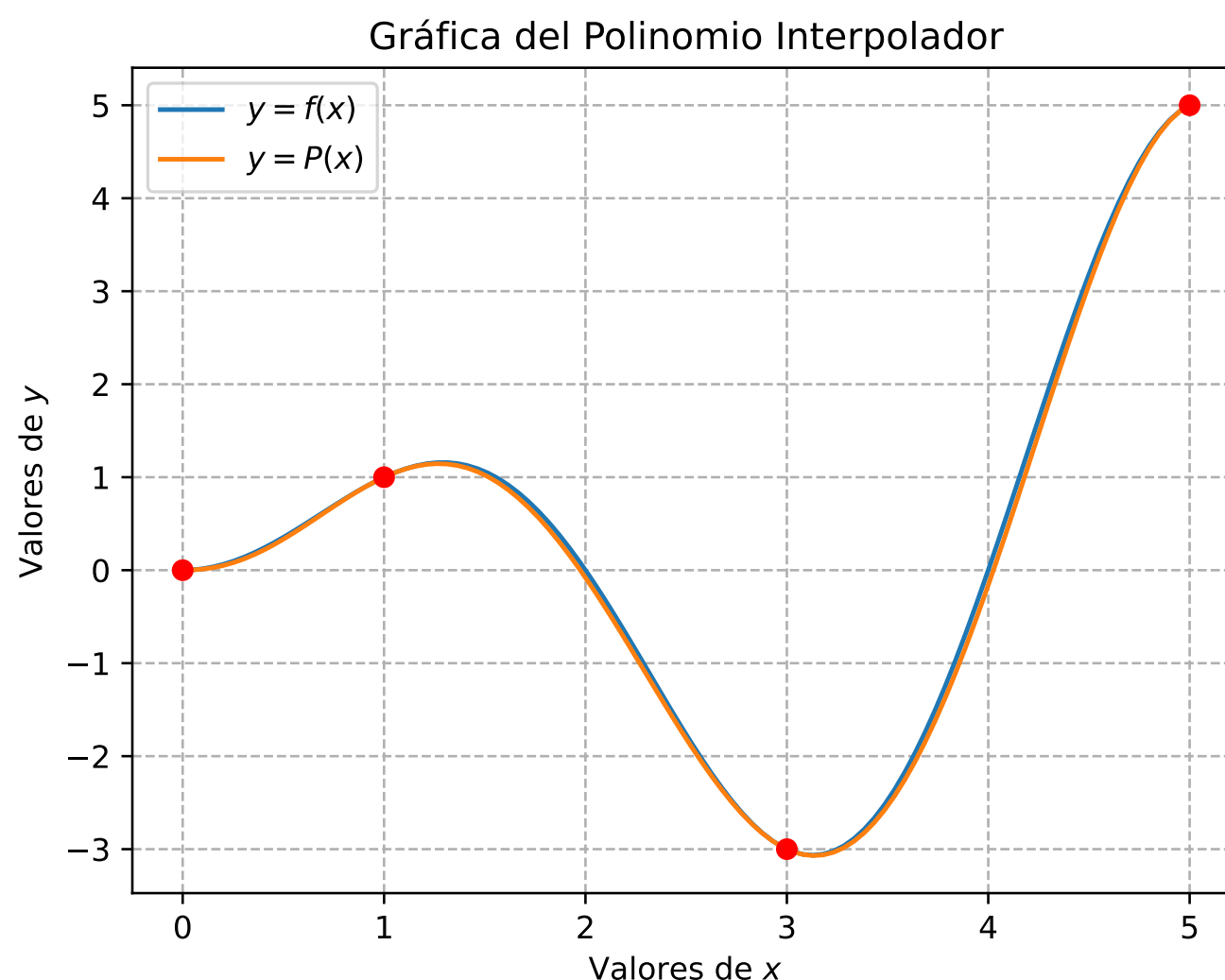


Figura 4.15: Gráfico del Polinomio Interpolador y la Función

Conclusiones

Este método es muy extenso, pero nos entrega un resultado muy bueno de la aproximación de la función. claro esta que solo se puede

usar cuando se conoce la función que da origen a los puntos de interpolación.

4.9 Taller de Interpolación

4.9.1 Física

1. Velocidad de un Objeto en Caída Libre.

- **Descripción:** Un objeto se deja caer desde una altura. Se mide su posición a diferentes tiempos durante la caída. Se desea estimar la velocidad del objeto en instantes intermedios.
- **Obtención de datos:**
 - ❑ Se puede usar un teléfono inteligente para grabar la caída, y luego un programa de videoanálisis para obtener la posición en función del tiempo.
 - ❑ O utilizar sensores de movimiento conectados a un microprocesador para obtener datos precisos.
- **Ley/fenómeno relacionado:** Cinemática, aceleración constante (gravedad).
- **Objetivo de la interpolación:** Determinar la velocidad instantánea del objeto en puntos donde no se tiene medición directa.

2. Enfriamiento de un Líquido.

- **Descripción:** Se calienta un líquido y se deja enfriar. Se mide la temperatura del líquido a diferentes tiempos. Se busca estimar la temperatura en momentos intermedios.
- **Obtención de datos:**
 - ❑ Un termómetro estándar, y un cronómetro.
 - ❑ Un sensor de temperatura conectado a un microprocesador.
- **Ley/fenómeno relacionado:** Termodinámica, ley de enfriamiento de Newton.
- **Objetivo de la interpolación:** Crear una curva de enfriamiento continua para analizar la tasa de cambio de temperatura.

3. Oscilación de un Péndulo.

- **Descripción:** Se mide el ángulo de un péndulo en diferentes instantes de tiempo mientras oscila. Se busca estimar el ángulo en momentos intermedios.
- **Obtención de datos:**
 - Se puede usar un video y un transportador para medir el ángulo.
 - Sensores rotacionales.
- **Ley/fenómeno relacionado:** Movimiento armónico simple.
- **Objetivo de la interpolación:** Reconstruir la trayectoria del péndulo con mayor precisión.

4. Relación Fuerza-Extensión de un Resorte.

- a) **Descripción:** Se cuelgan diferentes pesos de un resorte y se mide la extensión del resorte para cada peso. Se desea determinar la extensión para pesos intermedios.
- b) **Obtención de datos:** Regla y pesas de diferentes masas.
- c) **Ley/fenómeno relacionado:** Ley de Hooke.
- d) **Objetivo de la interpolación:** Determinar la constante de elasticidad del resorte.

5. Carga y Descarga de un Condensador.

- **Descripción:** Se mide el voltaje a través de un condensador mientras se carga o descarga a través de una resistencia. Se busca estimar el voltaje en momentos intermedios.
- **Obtención de datos:**
 - Multímetro y cronómetro.
 - Sensores de voltaje, conectados a un microprocesador.
- **Ley/fenómeno relacionado:** Circuitos RC.
- **Objetivo de la interpolación:** Analizar la curva de carga o descarga del condensador.

6. Posición de un objeto "A" que persigue un objeto "B".

- **Descripción:** un objeto B se mueve a velocidad constante, y un objeto A , lo persigue, incrementando su velocidad de forma uniforme. Se toman datos de las posiciones de ambos objetos en el tiempo.

- **Obtención de datos:** Mediciones de posición con sensores de movimiento o videoanálisis.
- **Ley/fenómeno relacionado:** Cinemática (relación entre posición, velocidad y aceleración).
- **Objetivo de la interpolación:** comparar las dos curvas de posición, para estimar en que momento, el objeto "A" alcanzara al objeto "B".

7. Determinación de la velocidad instantánea.

- **Descripción:** Un objeto se mueve con velocidad variable. Se miden su posición en diferentes instantes de tiempo. Se busca determinar la velocidad del objeto en un instante específico entre las mediciones.
- **Obtención de datos:** Mediciones de posición con sensores de movimiento o videoanálisis.
- **Ley/fenómeno relacionado:** Cinemática (relación entre posición, velocidad y aceleración).
- **Objetivo de la interpolación:** Estimar la velocidad en instantes donde no se tienen mediciones directas.

8. Análisis de la temperatura en un material.

- **Descripción:** Se mide la temperatura en varios puntos de un material. Se desea conocer la distribución de temperatura en todo el material.
- **Obtención de datos:** Termopares o cámaras termográficas.
- **Ley/fenómeno relacionado:** Conducción de calor (ley de Fourier).
- **Objetivo de la interpolación:** Crear un mapa de temperatura continuo a partir de los datos discretos.

9. Estudio de la propagación de ondas.

- **Descripción:** Se miden las amplitudes de una onda en diferentes puntos del espacio. Se busca reconstruir la forma completa de la onda.
- **Obtención de datos:** Sensores de onda (micrófonos, antenas, etc.).

- **Ley/fenómeno relacionado:** Ondas (superposición, interferencia, difracción).
- **Objetivo de la interpolación:** Obtener una representación continua de la onda para su análisis.

10. Análisis de la trayectoria de un proyectil.

- **Descripción:** Se mide la posición de un proyectil en varios puntos de su trayectoria. Se quiere determinar su posición en puntos intermedios.
- **Obtención de datos:** Videoanálisis, radares.
- **Ley/fenómeno relacionado:** Mecánica clásica (movimiento parabólico).
- **Objetivo de la interpolación:** Reconstruir la trayectoria completa del proyectil.

11. Variación de la presión atmosférica con la altitud.

- **Descripción:** Se mide la presión atmosférica a diferentes altitudes. Se quiere conocer la presión en altitudes intermedias.
- **Obtención de datos:** Barómetros en globos sonda o aviones.
- **Ley/fenómeno relacionado:** Estática de fluidos (variación de la presión con la profundidad).
- **Objetivo de la interpolación:** Crear un modelo de la variación de la presión con la altitud.

12. Calibración de un sensor.

- **Descripción:** se toman datos de un sensor a diferentes medidas estandarizadas, y se busca tener una función que transforme la lectura del sensor a la medida estandarizada.
- **Obtención de datos:** Mediante comparación entre el sensor evaluado y un aparato de medición estandarizado.
- **Ley/fenómeno relacionado:** Metrología.
- **Objetivo de la interpolación:** crear una función de transformación entre los valores de medición del sensor y los valores reales.

13. Análisis del Movimiento de un Balón de Fútbol.

- **Descripción:** Se graba un video de un balón de fútbol pateado, y se sigue su trayectoria. Se quiere determinar la velocidad y aceleración del balón en diferentes puntos de su vuelo.
- **Obtención de datos:**
 - Grabación de video del lanzamiento del balón.
 - Uso de software de videoanálisis (como Tracker) para obtener la posición del balón en cada fotograma.
- **Ley/fenómeno relacionado:** Cinemática, movimiento parabólico, resistencia del aire.
- **Objetivo de la interpolación:** Determinar la velocidad y aceleración instantáneas, y modelar la trayectoria del balón.

14. Estudio de la Oscilación de un Columpio.

- **Descripción:** Se graba un video de un columpio en movimiento. Se busca analizar el movimiento periódico y determinar el período de oscilación.
- **Obtención de datos:**
 - Grabación de video del columpio.
 - Medición del ángulo del columpio en función del tiempo mediante videoanálisis.
- **Ley/fenómeno relacionado:** Movimiento armónico simple, péndulo.
- **Objetivo de la interpolación:** Crear una función que describa el cambio del ángulo en el tiempo, para luego poder determinar el periodo.

15. Análisis del Movimiento de un Carro en una Rampa.

- **Descripción:** Se graba un video de un carro que baja por una rampa. Se busca analizar el movimiento acelerado y determinar la aceleración del carro.
- **Obtención de datos:**
 - Grabación de video del carro en la rampa.
 - Medición de la posición del carro en función del tiempo.
- **Ley/fenómeno relacionado:** Cinemática, movimiento uniformemente acelerado.

- **Objetivo de la interpolación:** Determinar la aceleración constante del carro y modelar su movimiento.

16. Estudio de la Trayectoria de una Pelota de Baloncesto.

- **Descripción:** Se graba un video de un lanzamiento de baloncesto. Se busca analizar la trayectoria de la pelota y determinar la velocidad inicial y el ángulo de lanzamiento.
- **Obtención de datos:**
 - Grabación de video del lanzamiento.
 - Medición de la posición de la pelota en cada fotograma.
- **Ley/fenómeno relacionado:** Cinemática, movimiento parabólico.
- **Objetivo de la interpolación:** Reconstruir la trayectoria completa de la pelota y determinar parámetros iniciales.

17. Análisis de la Caída de Diferentes Objetos.

- **Descripción:** Se graba un video de la caída de objetos de diferentes formas y pesos. Se busca comparar sus velocidades de caída y analizar el efecto de la resistencia del aire.
- **Obtención de datos:**
 - Grabación de video de la caída de los objetos.
 - Medición de la posición de los objetos en función del tiempo.
- **Ley/fenómeno relacionado:** Cinemática, caída libre, resistencia del aire.
- **Objetivo de la interpolación:** Poder comparar la variación de la velocidad, de los diferentes objetos, a medida que caen.

18. El movimiento de un liquido viscoso que cae.

- **Descripción:** Se graba la caída de una canica dentro de un tubo de un liquido con alta viscosidad, y se evalúa la velocidad de caída.
- **Obtención de datos:** Grabación de video del experimento, y medición de los cambios de posición de la canica, mientras cae.
- **Ley/fenómeno relacionado:** Dinámica de fluidos, viscosidad.
- **Objetivo de la interpolación:** Poder determinar la velocidad, y si esta varía o no durante la caída, para obtener conclusiones acerca de la viscosidad.

19. El movimiento de una persona corriendo.

- **Descripción:** Se graba a una persona corriendo en línea recta, y se evalúa los cambios de velocidad.
- **Obtención de datos:** Grabación de video de la persona corriendo, y medición de los cambios de posición a través del tiempo.
- **Ley/fenómeno relacionado:** Cinemática.
- **Objetivo de la interpolación:** Generar las gráficas de posición vs tiempo, y velocidad vs tiempo, del corredor.

20. La colisión de dos objetos.

- **Descripción:** Se graba la colisión de dos canicas o carros, y se evalúa la variación de la velocidad, antes, durante y después del choque.
- **Obtención de datos:** Grabación de video de la colisión, y medición de los cambios de posición de los objetos a través del tiempo.
- **Ley/fenómeno relacionado:** Dinámica, colisiones, conservación del momentum.
- **Objetivo de la interpolación:** poder determinar como varía la velocidad de los objetos, durante el tiempo que dura la colisión, y así poder obtener el valor aproximado de la fuerza de choque.

4.9.2 Matemáticas

1. Aproximación de funciones complejas.

- **Descripción:** Se conocen los valores de una función compleja en ciertos puntos del plano complejo. Se busca estimar su valor en otros puntos.
- **Obtención de datos:** Evaluación directa de la función en puntos específicos.
- **Ley/fenómeno relacionado:** Análisis complejo.
- **Objetivo de la interpolación:** Extender el dominio de la función a partir de valores conocidos.

2. Estimación de áreas bajo curvas.

- **Descripción:** Se tienen datos discretos que representan una curva. Se busca estimar el área bajo la curva.
- **Obtención de datos:** Mediciones o evaluaciones de funciones.
- **Ley/fenómeno relacionado:** Cálculo integral.
- **Objetivo de la interpolación:** Aproximar la integral definida de la función.

3. Resolución numérica de ecuaciones diferenciales.

- **Descripción:** Se busca aproximar la solución de una ecuación diferencial a partir de valores iniciales y puntos intermedios.
- **Obtención de datos:** Métodos numéricos como Euler o Runge-Kutta.
- **Ley/fenómeno relacionado:** Ecuaciones diferenciales.
- **Objetivo de la interpolación:** Obtener una solución aproximada de la ecuación diferencial.

4. Generación de curvas suaves en diseño gráfico.

- **Descripción:** Se tienen puntos de control que definen una curva. Se busca generar una curva suave que pase por estos puntos.
- **Obtención de datos:** Puntos de control definidos por el diseñador.
- **Ley/fenómeno relacionado:** Geometría computacional (curvas Bézier, splines).
- **Objetivo de la interpolación:** Crear curvas estéticamente agradables para diseño gráfico.

5. Predicción de series temporales.

- **Descripción:** Se tienen datos de una variable que varía con el tiempo. Se busca predecir valores futuros de la variable.
- **Obtención de datos:** Mediciones de la variable en diferentes instantes de tiempo.
- **Ley/fenómeno relacionado:** Análisis de series temporales.
- **Objetivo de la interpolación:** Estimar valores futuros a partir de datos pasados.

6. Procesamiento de imágenes.

- **Descripción:** Aumento de resolución de imágenes digitales.
- **Obtención de datos:** Píxeles de la imagen original.
- **Ley/fenómeno relacionado:** procesamiento digital de imágenes.
- **Objetivo de la interpolación:** Aumentar la cantidad de píxeles en una imagen existente de manera que no se vea pixelizada.

7. Aproximación del Seno/Coseno mediante Triángulos Rectángulos.

- **Descripción:** Los estudiantes construyen triángulos rectángulos variando un ángulo agudo y miden la longitud del cateto opuesto/adyacente y la hipotenusa. Usan estos datos para estimar los valores del seno y coseno del ángulo y los comparan con los valores teóricos.
- **Obtención de datos:** Mediciones con regla y transportador. Usando Geogebra la construcción se realiza con herramientas de geometría.
- **Ley/fenómeno relacionado:** Trigonometría, funciones trigonométricas.
- **Objetivo de la interpolación:** Aproximar las funciones seno y coseno mediante un conjunto de datos experimentales y contrastar con su representación analítica.

8. Determinación de la Relación entre el Área y el Radio de un Círculo.

- **Descripción:** Los estudiantes dibujan círculos de diferentes radios y miden sus áreas (por ejemplo, dividiéndolos en sectores y aproximándolos a triángulos). Utilizan interpolación para encontrar la relación entre el área y el radio.
- **Obtención de datos:** Mediciones con regla y compás. Usando geogebra para determinar el área.
- **Ley/fenómeno relacionado:** Geometría, área del círculo.
- **Objetivo de la interpolación:** Obtener la fórmula empírica del área del círculo y compararla con la fórmula conocida ($A = \pi r^2$)

9. Estudio de la Convergencia de una Serie Geométrica.

- **Descripción:** Los estudiantes generan los primeros términos de una serie geométrica convergente (por ejemplo, $1/2$, $1/4$, $1/8$, ...) y calculan las sumas parciales. Utilizan interpolación para estimar la suma límite de la serie.
- **Obtención de datos:** Cálculos manuales o con calculadora/hoja de cálculo.
- **Ley/fenómeno relacionado:** Series geométricas, convergencia de series.
- **Objetivo de la interpolación:** Visualizar cómo las sumas parciales se acercan a un valor límite y estimar ese valor.

10. Variación del Perímetro de un Polígono Inscrito en un Círculo.

- **Descripción:** Los estudiantes inscriben polígonos regulares de diferentes números de lados en un círculo y miden sus perímetros. Utilizan interpolación para estimar el perímetro del círculo y aproximar el valor de π .
- **Obtención de datos:** Dibujos con regla y compás. Usando geogebra para la construcción, y calculo del perímetro.
- **Ley/fenómeno relacionado:** Geometría, aproximación de π .
- **Objetivo de la interpolación:** Observar cómo el perímetro del polígono se acerca a la circunferencia del círculo a medida que aumenta el número de lados.

11. Análisis de la Función Exponencial mediante Población Simulada.

- **Descripción:** Mediante el uso de geogebra, Los estudiantes simulan una población que crece exponencialmente (por ejemplo, bacterias que se duplican cada hora) y registran el tamaño de la población a intervalos regulares de tiempo. Utilizan interpolación para modelar el crecimiento y estimar la tasa de crecimiento.
- **Obtención de datos:** Simulación computacional.
- **Ley/fenómeno relacionado:** Funciones exponenciales, crecimiento exponencial.
- **Objetivo de la interpolación:** Obtener la fórmula empírica del crecimiento exponencial y compararla con la fórmula general ($y = a \cdot e^{(kt)}$).

12. Obtención de una curva de Bézier.

- **Descripción:** usando geogebra los estudiantes colocan 4 puntos de control, y luego generan la curva de Bezier. Luego varían la posición de los puntos, y analizan el cambio de la forma de la curva.
- **Obtención de datos:** datos generados por geogebra.
- **Ley/fenómeno relacionado:** geometría computacional, curvas de Bezier.
- **Objetivo de la interpolación:** analizar los efectos de cambiar los puntos de control.

4.9.3 Biología

1. Crecimiento de una población bacteriana.

- **Descripción:** Se mide el número de bacterias en una población en intervalos de tiempo regulares. Se desea estimar el número de bacterias en momentos intermedios.
- **Obtención de datos:** Conteo de colonias en placas de Petri, o medición de la densidad óptica.
- **Ley/fenómeno relacionado:** Cinética de crecimiento bacteriano (fase exponencial, fase estacionaria).
- **Objetivo de la interpolación:** Predecir el tamaño de la población entre mediciones y modelar la curva de crecimiento.

2. Concentración de glucosa en sangre.

- **Descripción:** Se toman muestras de sangre de un paciente cada hora y se mide la concentración de glucosa. Se busca estimar la concentración en momentos intermedios.
- **Obtención de datos:** Análisis de muestras de sangre con glucómetros.
- **Ley/fenómeno relacionado:** Metabolismo de la glucosa, homeostasis.
- **Objetivo de la interpolación:** Crear una curva continua de la concentración de glucosa para evaluar la respuesta del paciente a tratamientos o alimentos.

3. Tasa de fotosíntesis.

- **Descripción:** Se mide la cantidad de oxígeno producido por una planta a diferentes intensidades de luz. Se desea estimar la tasa de fotosíntesis a intensidades de luz intermedias.
- **Obtención de datos:** Medición del oxígeno disuelto en agua o uso de cámaras de intercambio de gases.
- **Ley/fenómeno relacionado:** Fotosíntesis, relación entre luz y producción de oxígeno.
- **Objetivo de la interpolación:** Determinar la respuesta de la planta a variaciones en la intensidad de luz.

4. Curva de crecimiento de un organismo.

- **Descripción:** Se mide el tamaño o peso de un organismo (planta, animal) a lo largo de su desarrollo. Se quiere estimar el tamaño o peso en momentos intermedios.
- **Obtención de datos:** Mediciones directas del tamaño o peso.
- **Ley/fenómeno relacionado:** Biología del desarrollo, patrones de crecimiento.
- **Objetivo de la interpolación:** Generar una curva de crecimiento suave para visualizar y analizar el desarrollo del organismo.

5. Variación de la concentración de una hormona.

- **Descripción:** Se miden los niveles de una hormona en la sangre de un animal a lo largo del tiempo. Se quiere estimar los niveles hormonales en momentos entre las mediciones.
- **Obtención de datos:** Análisis de sangre mediante inmunoensayos.
- **Ley/fenómeno relacionado:** Endocrinología, ritmos circadianos.
- **Objetivo de la interpolación:** Estudiar la dinámica de la secreción hormonal.

6. Dilatación de la pupila.

- **Descripción:** Se mide el diámetro de la pupila de un animal en respuesta a diferentes intensidades de luz. Se busca estimar el tamaño de la pupila con luces de intensidades intermedias.

- **Obtención de datos:** Mediciones de fotografías tomadas del ojo del espécimen evaluado.
- **Ley/fenómeno relacionado:** Fisiología, respuesta del sistema nervioso a la luz.
- **Objetivo de la interpolación:** Determinar la respuesta fisiológica en el organismo estudiado.

7. Densidad poblacional de una especie.

- **Descripción:** Se evalúa la densidad de individuos de alguna especie de animal o planta en un área determinada, y se repite la medición en áreas contiguas, o variando un parámetro ambiental (ej: altitud).
- **Obtención de datos:** Mediante conteo directo, o evaluaciones de video o fotografía aérea.
- **Ley/fenómeno relacionado:** Ecología de poblaciones.
- **Objetivo de la interpolación:** generar mapas de calor donde se aprecie la variación de la densidad poblacional.