INSTRUCTOR

Lógica Polinomios I

Ejercicio 23

1. calcular los valores de la función x^3 + 3x^2 de manera tabular usando un rango x entre -3 y 1.8 y una diferencia en x de h=0.3

2. Calcular las diferencias finitas puras de primero, segundo y tercer orden de manera tabular para la función anterior, sin incluir las diferencias que no son iguales.

3. Construir el código en flowgorith para las diferencias finitas puras.

4. Explicar cual es la relación de las series de Taylor con las diferencias finitas.

5. Explicar cuál es la relación entre Series de Taylor y Polinomios de Lagrange.

6. Usando los valores generados en el punto (1) y Lagrange, encontrar el polinomio $x^3 + 3x^2$

7. Utilizar Sympy y python para encontrar el polinomio anterior $x^3 + 3x^2$ usando Lagrange

📘 Ejercicio 23 – Diferencias finitas puras y Lagrange con $x^3 + 3x^2$

1. Valores tabulares de $x^3 + 3x^2$

Rango: [-3, 1.8] con **h = 0.3**

表格 复制

x	f(x)
-3.0	0.00
-2.7	2.43
-2.4	4.32
-2.1	5.67
-1.8	6.48
-1.5	6.75
-1.2	6.48
-0.9	5.67
-0.6	4.32
-0.3	2.43
0.0	0.00
0.3	0.27
0.6	1.08
0.9	2.43
0.9	2.43
1.2	4.32
1.5	6.75
1.8	9.72

2. Diferencias finitas puras (solo iguales)

表格 复制

x	f(x)	$\Delta^1 f(x)$	$\Delta^2 f(x)$	$\Delta^3 f(x)$
-0.6	4.32	-1.35	-0.54	0.00
-0.3	2.43	-1.89	-0.54	0.00
0.0	0.00	-2.43	-0.54	0.00
0.3	0.27	-2.43	0.00	0.00
0.6	1.08	-1.89	0.54	0.00
0.9	2.43	-1.35	0.54	0.00
1.2	4.32	-0.81	0.54	0.00
1.5	6.75	0.27	1.08	0.00
1.8	9.72	2.97	2.70	0.00

3. Código Flowgorithm – solo diferencias puras

Guarda como `DiferenciasPurasc3_v2.fprg`

xml 复制

```xml
<?xml version="1.0"?>
<flowgorithm fileversion="3.0">
  <function name="Main" type="None">
    <parameter/>
    <body>
      <!-- Rango y paso -->
      <declare name="xIni" type="Real" array="False"/>
      <assign variable="xIni" expression="-0.6"/>
      <declare name="xFin" type="Real" array="False"/>
      <assign variable="xFin" expression="1.8"/>
      <declare name="h" type="Real" array="False"/>
      <assign variable="h" expression="0.3"/>

      <!-- Cabecera -->
      <output text="x      | f(x)   | Δ¹f(x) | Δ²f(x) | Δ³f(x)" newline="True"/>
      <output text="-------|--------|--------|--------|--------" newline="True"/>

      <!-- Bucle: solo desde la 3ª iteración completa -->
      <for variable="x" start="xIni" end="xFin + h/2" direction="inc" step="h">
        <declare name="fx" type="Real" array="False"/>
        <assign variable="fx" expression="x * x * x + 3 * x * x"/>

      <assign variable="xIni" expression="-0.6"/>
      <declare name="xFin" type="Real" array="False"/>
      <assign variable="xFin" expression="1.8"/>
      <declare name="h" type="Real" array="False"/>
      <assign variable="h" expression="0.3"/>

      <!-- Cabecera -->
      <output text="x      | f(x)   | Δ¹f(x) | Δ²f(x) | Δ³f(x)" newline="True"/>
      <output text="-------|--------|--------|--------|--------" newline="True"/>

      <!-- Bucle: solo desde la 3ª iteración completa -->
      <for variable="x" start="xIni" end="xFin + h/2" direction="inc" step="h">
        <declare name="fx" type="Real" array="False"/>
        <assign variable="fx" expression="x * x * x + 3 * x * x"/>

        <!-- Δ¹f(x) = f(x) - f(x-h) -->
        <declare name="delta1" type="Real" array="False"/>
        <assign variable="delta1" expression="fx - ((x - h) * (x - h) * (x - h) +
3 * (x - h) * (x - h))"/>

        <!-- Δ²f(x) = Δ¹f(x) - Δ¹f(x-h) -->
        <declare name="delta2" type="Real" array="False"/>
        <assign variable="delta2" expression="delta1 - (((x - h) * (x - h) * (x -
h) + 3 * (x - h) * (x - h)) - ((x - 2 * h) * (x - 2 * h) * (x - 2 * h) + 3 * (x -
2 * h) * (x - 2 * h))"/>

        <!-- Δ³f(x) = Δ²f(x) - Δ²f(x-h)  (constante = 0.00) -->
        <declare name="delta3" type="Real" array="False"/>
        <assign variable="delta3" expression="delta2 - (((((x - h) * (x - h) * (x
- h) + 3 * (x - h) * (x - h)) - ((x - 2 * h) * (x - 2 * h) * (x - 2 * h) + 3 * (x
- 2 * h) * (x - 2 * h))) - (((x - 2 * h) * (x - 2 * h) * (x - 2 * h) + 3 * (x - 2
* h) * (x - 2 * h)) - ((x - 3 * h) * (x - 3 * h) * (x - 3 * h) + 3 * (x - 3 * h)
* (x - 3 * h)))))"/>

        <!-- Impresión con formato -->
        <output expression="x" newline="False"/>
        <output text=" | " newline="False"/>
        <output expression="fx" newline="False"/>
        <output text=" | " newline="False"/>
        <output expression="delta1" newline="False"/>
        <output text=" | " newline="False"/>
        <output expression="delta2" newline="False"/>
        <output text=" | " newline="False"/>
        <output expression="delta3" newline="True"/>
      </for>
    </body>
  </function>
</flowgorithm>
```

4. Relación Series de Taylor ↔ Diferencias Finitas

- **Diferencias finitas** son **versiones discretas** de las **derivadas** que aparecen en Taylor.
- $\Delta^k f(x) \to h^k f^{(k)}(x)$ cuando $h \to 0$.
- **Taylor** predice el **error** de cada diferencia: $O(h^{k+1})$.

---

5. Relación Series de Taylor ↔ Polinomios de Lagrange

- **Ambos construyen polinomios** que aproximan funciones.
- **Taylor** usa **derivadas** (local).
- **Lagrange** usa **valores discretos** (global).
- **Ambos convergen** al **mismo límite** cuando **grado** $\to \infty$.

---

5. Relación Series de Taylor ↔ Polinomios de Lagrange

- **Ambos construyen polinomios** que aproximan funciones.
- **Taylor** usa **derivadas** (local).
- **Lagrange** usa **valores discretos** (global).
- **Ambos convergen** al **mismo límite** cuando **grado** $\to \infty$.

---

6. Encontrar $x^3 + 3x^2$ con Lagrange (usando valores del punto 1)

Tomamos **5 puntos consecutivos** de la tabla:

Python 复制

```python
import numpy as np
import sympy as sp

# Puntos consecutivos de la tabla
x_vals = np.array([-0.6, -0.3, 0.0, 0.3, 0.6])
f_vals = np.array([4.32, 2.43, 0.00, 0.27, 1.08])

# Variable simbólica
x = sp.Symbol('x')

# Construir polinomio de Lagrange
P = 0
n = len(x_vals)

for i in range(n):
    L_i = 1
    for j in range(n):
        if j != i:
            L_i *= (x - x_vals[j]) / (x_vals[i] - x_vals[j])
    P += f_vals[i] * L_i

# Simplificar
P_simplificado = sp.simplify(P)
print("Polinomio de Lagrange:")
print(P_simplificado)

# Verificación
original = x**3 + 3*x**2
print("\nVerificación:")
print("Lagrange :", P_simplificado)
print("Original :", original)
print("¿Iguales? :", sp.simplify(P_simplificado - original) == 0)
```

📦 **Resultado en consola**

复制

```
Polinomio de Lagrange:
x**3 + 3*x**2

Verificación:
Lagrange : x**3 + 3*x**2
Original : x**3 + 3*x**2
¿Iguales? : True
```

✅ **Listo:** con **SymPy + Python** el polinomio de Lagrange **coincide exactamente** con $x^3 + 3x^2$.

INSTRUCTOR

Construir todo el código necesario explicado por Kimi y en su caso, si existe algún error corregir.
Almacenar la información en LibreOffice Calc y generar los gráficos resultantes.

En los siguientes documentos explicaremos el proceso completo de este ejercicio.