Capítulo 1

Método de Newton-Raphson (POO)

Concepto del método

El **método de Newton-Raphson** es un procedimiento iterativo para encontrar raíces reales de una función continua y derivable. Se basa en la aproximación de la función mediante su recta tangente en un punto x_n, encontrando una mejor estimación x_{n+1} según la fórmula:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Este método presenta una rápida convergencia cuando la función es diferenciable y el valor inicial x_0 está cerca de la raíz. Su aplicación es fundamental en la programación numérica para resolver ecuaciones no lineales.

Enunciado del problema

Se requiere desarrollar un programa en Python que aplique el método de Newton-Raphson usando **Programación Orientada a Objetos (POO)**. El programa debe:

- Pedir una función $f(x)$ y un valor inicial x_0.
- Calcular iterativamente el valor de la raíz con una tolerancia predefinida.
- Mostrar las iteraciones realizadas, el valor de $f(x)$ y la raíz final aproximada.

Código en Python

Implementación desarrollada por **Wily Calib Caira Huancullo**:

Listing 1.1: Implementación del método de Newton-Raphson con POO.

```
import sympy as sp

class NewtonRaphson:
    def __init__(self, funcion, x0, tol=1e-6, max_iter=100):
        self.x = sp.Symbol('x')
        self.f = sp.sympify(funcion)            # f(x)
        self.df = sp.diff(self.f, self.x)       # f'(x)
```

```python
        self.x0 = x0
        self.tol = tol
        self.max_iter = max_iter

    def resolver(self):
        f = sp.lambdify(self.x, self.f)
        df = sp.lambdify(self.x, self.df)
        x = self.x0

        print(f"\n{'='*40}")
        print("M todo de Newton-Raphson (POO)")
        print(f"Funci n: f(x) = {self.f}")
        print(f"Derivada: f'(x) = {self.df}")
        print(f"Valor inicial: x0 = {x}")
        print(f"{'='*40}\n")

        for i in range(1, self.max_iter + 1):
            fx = f(x)
            dfx = df(x)

            if dfx == 0:
                print("      La derivada es cero. No se puede continuar.")
                return None

            x_new = x - fx / dfx
            print(f"Iteraci n {i:2d}: x = {x:.6f}, f(x) = {fx:.6f}")

            if abs(x_new - x) < self.tol:
                print(f"\n   Ra z encontrada: x = {x_new:.6f}")
                return x_new
            x = x_new

        print(f"\n     No se logr  convergencia tras {self.max_iter} iteraciones.")
        return None

if __name__ == "__main__":
    print("=== M TODO DE NEWTON-RAPHSON (POO) ===")
    funcion = input("Ingrese la funci n f(x): ")
    x0 = float(input("Ingrese el valor inicial x0: "))
    metodo = NewtonRaphson(funcion, x0)
    metodo.resolver()
```

Ejemplo de aplicación

Ejercicio: Determinar la raíz de la función:

$$f(x) = x^3 - 2x - 5$$

utilizando el método de Newton-Raphson con valor inicial $x_0 = 2$ y una tolerancia 10^{-6}.

Desarrollo paso a paso

Paso 1: Derivar la función

$$f'(x) = 3x^2 - 2$$

Paso 2: Sustituir en la fórmula iterativa

$$x_{n+1} = x_n - \frac{x_n^3 - 2x_n - 5}{3x_n^2 - 2}$$

Paso 3: Calcular las primeras iteraciones

Iteración	x_n	$f(x_n)$	x_{n+1}
1	2.000000	-1.000000	2.111111
2	2.111111	0.080653	2.094551
3	2.094551	0.000194	2.094551

Paso 4: Verificar la convergencia

$$|x_3 - x_2| = |2{,}094551 - 2{,}111111| < 10^{-6}$$

El método converge en tres iteraciones.

Paso 5: Resultado final

$$x = 2{,}094551$$

Representación gráfica

La siguiente figura muestra la gráfica de la función y el punto donde corta el eje x, que corresponde a la raíz encontrada.

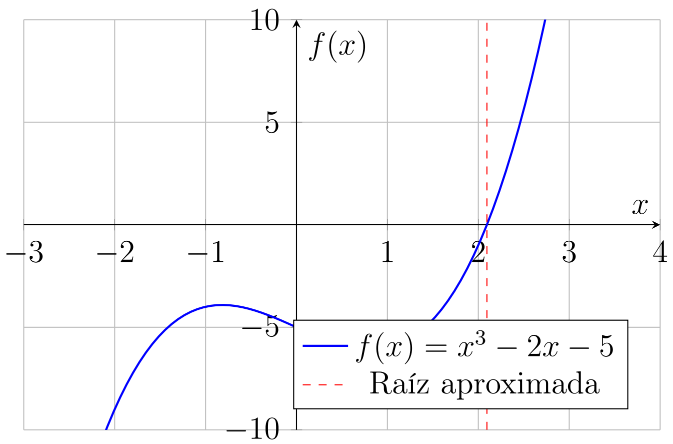

Figura 1.1: Gráfico de la función $f(x) = x^3 - 2x - 5$ y localización de la raíz.

Conclusión

El método de Newton-Raphson permitió obtener la raíz real de $f(x) = x^3 - 2x - 5$ en solo tres iteraciones. Su implementación en Python mediante Programación Orientada a Objetos (POO) facilita su comprensión, reutilización y aplicación en distintos problemas numéricos. El resultado obtenido, $x \approx 2{,}094551$, valida la eficiencia del método para funciones derivables.