Universidad Nacional del Altiplano
Facultad de Ingeniería Estadística e Informática

Docente: Ing. Fred Torres Cruz
Estudiante: Luis Angel Quenaya Loza
Código: 241411

Actividad N°05

Método de Newton-Raphson en Python

Descripción

El método de Newton-Raphson es un procedimiento iterativo que permite obtener una aproximación de las raíces reales de una función continua y derivable. Su fundamento se basa en el desarrollo del polinomio de Taylor de primer orden alrededor de un punto inicial x_0, con la siguiente fórmula recursiva:

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

En cada iteración se utiliza la pendiente de la tangente en el punto x_n para aproximarse progresivamente a la raíz. Si la función y su derivada son continuas, y el punto inicial está cerca de la raíz, la convergencia suele ser rápida y cuadrática.

Importancia de la visualización

Antes de aplicar el método, es recomendable graficar la función $f(x)$ en un intervalo adecuado. Esto permite identificar las zonas donde cruza el eje x, estimar una raíz posible y elegir un valor inicial x_0 apropiado. De esta manera, se incrementan las probabilidades de que el método converja correctamente.

Restricciones y limitaciones

- No se debe aplicar cuando $f'(x) = 0$, ya que genera una indeterminación.
- Si el punto inicial está lejos de la raíz, la convergencia no está garantizada.
- En raíces múltiples, la convergencia puede ser lenta o inestable.
- Se requiere que tanto $f(x)$ como $f'(x)$ sean continuas cerca de la raíz buscada.

Entrada

- Función $f(x)$ ingresada por el usuario.
- Límite inferior y superior del intervalo para graficar.
- Valor inicial x_0 para iniciar el método.

Salida

- Gráfico de la función $f(x)$.
- Raíz aproximada encontrada.
- Número de iteraciones necesarias para alcanzar la tolerancia establecida.

Código en Python

```python
import numpy as np
import matplotlib.pyplot as plt
import time

ecuacion = input("Ingrese la funci n f(x): ")

def f(x):
    return eval(ecuacion)

def derivada(x):
    h = 1e-6
    return (f(x + h) - f(x - h)) / (2 * h)

a = float(input("Ingrese el l mite inferior del eje x: "))
b = float(input("Ingrese el l mite superior del eje x: "))

x_vals = np.linspace(a, b, 400)
y_vals = f(x_vals)

plt.figure(figsize=(8, 5))
plt.plot(x_vals, y_vals, 'b', label=f"f(x) = {ecuacion}")
plt.axhline(0, color='black', linestyle='--')
plt.axvline(0, color='black', linestyle='--')
plt.title("Gr fico de la funci n ingresada")
plt.xlabel("x")
plt.ylabel("f(x)")
plt.legend()
plt.grid(True)
```

```
plt.show(block=False)
time.sleep(3)
plt.close()

x = float(input("Ingrese el valor inicial aproximado: "))
tolerancia = 1e-6
iter_max = 100

for i in range(iter_max):
    fx = f(x)
    dfx = derivada(x)
    if dfx == 0:
        print(f"La derivada es cero en x = {x}. No se puede continuar.")
        break
    x_nuevo = x - fx / dfx
    if abs(x_nuevo - x) < tolerancia:
        print(f"\nRa z encontrada: {x_nuevo:.6f}")
        print(f"N mero de iteraciones: {i + 1}")
        break
    x = x_nuevo
else:
    print(f"\nNo se logr  la convergencia despu s de {iter_max} iteraciones.")
    print(f" ltimo  valor aproximado: {x:.6f}")
```

Ejemplo de ejecución

```
Ingrese la función f(x): x**2 - x - 4
Ingrese el límite inferior del eje x: 1
Ingrese el límite superior del eje x: 5
Ingrese el valor inicial aproximado: 1
Raíz encontrada: 2.561553
Número de iteraciones: 6
```

Resultados gráficos

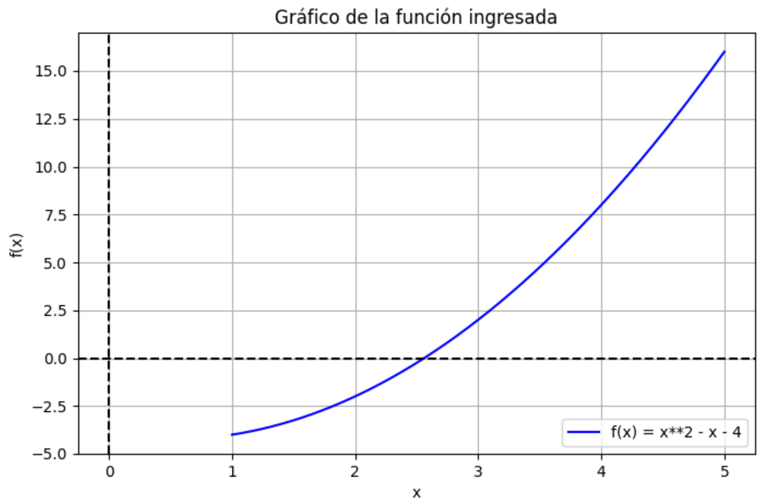

Figura 1: Gráfico de la función ingresada en Python.