Universidad Nacional del Altiplano
Facultad de Ingeniería Estadística e Informática

Docente:

Ing. Torres Cruz Fred

Alumno:

Ticona Miramira Roberto Angel

Trabajo 6 - Método de Bisección

» Descripción

El método de bisección es un procedimiento iterativo empleado para encontrar una raíz real de una función continua $f(x)$ dentro de un intervalo cerrado $[a, b]$ en el cual la función cambia de signo, es decir, $f(a) \cdot f(b) < 0$. El método se fundamenta en el **teorema del valor intermedio**, que garantiza la existencia de al menos una raíz en dicho intervalo.

El algoritmo consiste en dividir repetidamente el intervalo a la mitad y seleccionar el subintervalo en el que se conserva el cambio de signo. En cada iteración, se calcula el punto medio:

$$c = \frac{a + b}{2}$$

y se evalúa el signo de $f(c)$. Si $f(a) \cdot f(c) < 0$, entonces la raíz se encuentra en el intervalo $[a, c]$; de lo contrario, en $[c, b]$. Este proceso se repite hasta que la longitud del intervalo sea menor que una tolerancia prefijada o hasta alcanzar un número máximo de iteraciones.

Importancia de la visualización previa

Antes de aplicar el método de bisección, es recomendable graficar la función $f(x)$ en un rango apropiado. Esto permite identificar los puntos donde la función cruza el eje x y seleccionar adecuadamente el intervalo $[a, b]$ donde ocurre el cambio de signo. De esta manera, se asegura que el método cumpla su condición de aplicabilidad y converja a una raíz real.

Ventajas y limitaciones

El método de bisección presenta las siguientes características:

- Es un método **seguro y garantizado**, siempre que $f(a) \cdot f(b) < 0$.
- No requiere derivadas, a diferencia de métodos como el de Newton-Raphson.
- Su convergencia es **lineal**, lo que lo hace más lento que otros métodos iterativos.
- No puede aplicarse si la función no cambia de signo en el intervalo inicial.

» Aplicación del método en Python

Se desarrolló en Python un programa que permite al usuario ingresar una función matemática y visualizar su gráfico en un intervalo definido. Esta etapa inicial facilita la elección del intervalo $[a, b]$ donde la función cambia de signo.

Una vez seleccionado el intervalo, el programa ejecuta el método de bisección iterativamente, mostrando en pantalla una tabla con los valores de a, b, c, $f(c)$ y el error de cada iteración. El

proceso finaliza cuando se cumple la condición de tolerancia o se alcanza el número máximo de iteraciones. Finalmente, se presenta la raíz aproximada encontrada.

Este enfoque combina el análisis gráfico con la interpretación numérica, permitiendo comprender mejor el comportamiento de la función y la estabilidad del método.

» Entrada

Una cadena de texto que representa una función matemática.

$$f(x) = x^3 - x - 1$$

» Salida

- Gráfica para evaluar el intervalo de búsqueda.
- Tabla con las iteraciones del método.
- Raíz aproximada y número de iteraciones realizadas.

» Restricciones

El método requiere que:

- $f(x)$ sea continua en el intervalo $[a, b]$.
- Se cumpla la condición $f(a) \cdot f(b) < 0$.

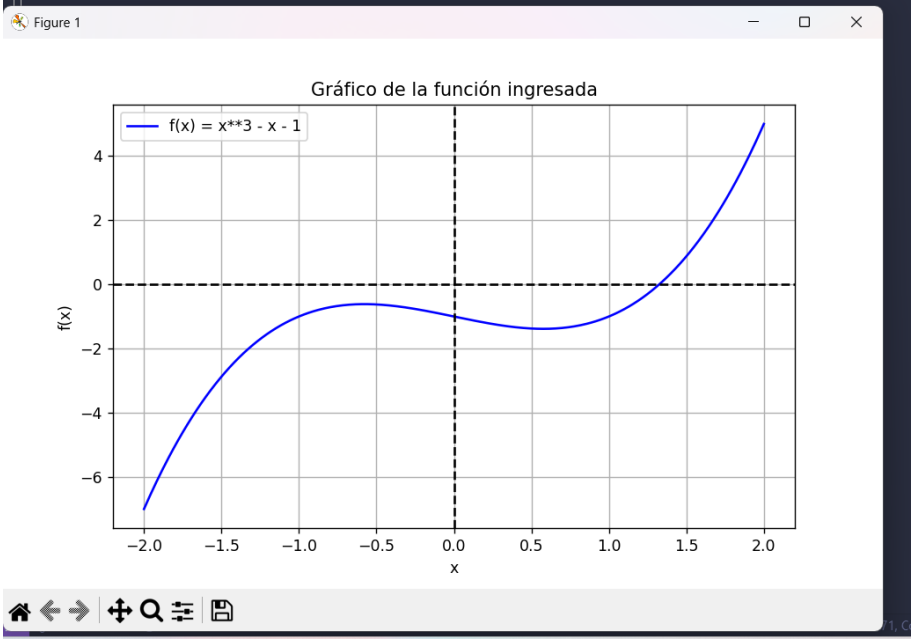
Código

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

# === 1. Ingreso de la función ===
func_str = input("Ingrese la función f(x): ")  # Ejemplo: x**3 - x - 1

# Definimos la función
def f(x):
    return eval(func_str, {"np": np, "x": x})

# === 2. Graficar la función ===
xmin = float(input("Ingrese el valor mínimo de x: "))
xmax = float(input("Ingrese el valor máximo de x: "))

x = np.linspace(xmin, xmax, 400)
y = f(x)

plt.figure(figsize=(8, 5))
plt.plot(x, y, label=f"f(x) = {func_str}", color='blue')
plt.axhline(0, color='black', linestyle='--')
plt.axvline(0, color='black', linestyle='--')
plt.title("Gráfico de la función ingresada")
plt.xlabel("x")
plt.ylabel("f(x)")
plt.legend()
plt.grid(True)
plt.show()

# === 3. Pregunta si desea aplicar el método de Bisección ===
op = input("¿Desea encontrar una raíz con el método de Bisección? (s/n): ").lower()

if op == "s":
```

```
    # === 4. Ingreso del intervalo inicial ===
    a = float(input("Ingrese el extremo izquierdo del intervalo (a): "))
    b = float(input("Ingrese el extremo derecho del intervalo (b): "))

    # Verificación del cambio de signo
    if f(a) * f(b) > 0:
        print("\n No hay cambio de signo en el intervalo [a, b]. Intente con otro intervalo.")
    else:
        # Parámetros del método
        tol = 1e-6
        max_iter = 100

        print("\nIteración |     a       |      b       |      c       |    f(c)      |   Error")
        print("-------------------------------------------------------------------------")

        for i in range(1, max_iter + 1):
            c = (a + b) / 2
            fc = f(c)
            error = abs(b - a) / 2

            print(f"{i:9d} | {a:10.6f} | {b:10.6f} | {c:10.6f} | {fc:10.6f} | {error:10.6f}")

            if abs(fc) < tol or error < tol:
                print(f"\n Raíz aproximada encontrada: {c:.6f}")
                print(f"Iteraciones realizadas: {i}")
                break

            if f(a) * fc < 0:
                b = c
            else:
                a = c
        else:
            print("\n No se alcanzó la convergencia después de", max_iter, "iteraciones.")
else:
    print("No se aplicó el método de Bisección.")
```

Ejecución

```
Ingrese la función f(x): x**3 - x - 1
Ingrese el valor mínimo de x: -2
Ingrese el valor máximo de x: 2
```




```
Ingrese la función f(x): x**3 - x - 1
Ingrese el valor mínimo de x: -2
Ingrese el valor máximo de x: 2
¿Desea encontrar una raíz con el método de Bisección? (s/n): s
Ingrese el extremo izquierdo del intervalo (a): 1
Ingrese el extremo derecho del intervalo (b): 2

Iteración |     a       |     b       |     c       |    f(c)     |   Error
-------------------------------------------------------------------------
        1 |   1.000000  |   2.000000  |   1.500000  |   0.875000  |   0.500000
        2 |   1.000000  |   1.500000  |   1.250000  |  -0.296875  |   0.250000
        3 |   1.250000  |   1.500000  |   1.375000  |   0.224609  |   0.125000
        4 |   1.250000  |   1.375000  |   1.312500  |  -0.051514  |   0.062500
        5 |   1.312500  |   1.375000  |   1.343750  |   0.082611  |   0.031250
        6 |   1.312500  |   1.343750  |   1.328125  |   0.014576  |   0.015625
        7 |   1.312500  |   1.328125  |   1.320312  |  -0.018711  |   0.007812
        8 |   1.320312  |   1.328125  |   1.324219  |  -0.002128  |   0.003906
        9 |   1.324219  |   1.328125  |   1.326172  |   0.006209  |   0.001953
       10 |   1.324219  |   1.326172  |   1.325195  |   0.002037  |   0.000977
       11 |   1.324219  |   1.325195  |   1.324707  |  -0.000047  |   0.000488
       12 |   1.324707  |   1.325195  |   1.324951  |   0.000995  |   0.000244
       13 |   1.324707  |   1.324951  |   1.324829  |   0.000474  |   0.000122
       14 |   1.324707  |   1.324829  |   1.324768  |   0.000214  |   0.000061
       15 |   1.324707  |   1.324768  |   1.324738  |   0.000084  |   0.000031
       16 |   1.324707  |   1.324738  |   1.324722  |   0.000018  |   0.000015
       17 |   1.324707  |   1.324722  |   1.324715  |  -0.000014  |   0.000008
       18 |   1.324715  |   1.324722  |   1.324718  |   0.000002  |   0.000004
       19 |   1.324715  |   1.324718  |   1.324717  |  -0.000006  |   0.000002
       20 |   1.324717  |   1.324718  |   1.324718  |  -0.000002  |   0.000001

✅ Raíz aproximada encontrada: 1.324718
Iteraciones realizadas: 20
```