Universidad Nacional del Altiplano de Puno

Escuela Profesional de Ingeniería Estadística e Informática

Método de Bisección

Definición, uso y código en Python

Estudiante: Wily Calib Caira Huancollo
Docente: Fred Cruz Torres
Curso: Programación Numérica

Puno - Perú
2025

Definición del Método de Bisección

El método de bisección es un procedimiento numérico utilizado para encontrar una raíz real de una función continua $f(x)$ dentro de un intervalo $[a, b]$. Se fundamenta en el Teorema del Valor Intermedio, el cual establece que si $f(a)$ y $f(b)$ tienen signos opuestos, entonces existe al menos una raíz en dicho intervalo. El método consiste en dividir repetidamente el intervalo a la mitad, evaluando la función en el punto medio, hasta obtener una aproximación suficientemente precisa de la raíz.

¿Para qué sirve?

Este método es útil cuando se desea encontrar soluciones aproximadas de ecuaciones no lineales donde no es posible resolver analíticamente. Es ampliamente usado en:

- Modelos matemáticos
- Problemas físicos y de ingeniería
- Métodos numéricos
- Simulación y análisis computacional

¿Cómo se usa?

1. Se selecciona un intervalo $[a, b]$ donde $f(a) \cdot f(b) < 0$.
2. Se calcula el punto medio $m = \frac{a+b}{2}$.
3. Se evalúa la función en m: $f(m)$.
4. Si el resultado es suficientemente cercano a cero, m es la raíz.
5. Si no, se reemplaza a o b dependiendo del signo de $f(m)$.
6. El proceso se repite hasta alcanzar la tolerancia deseada.

Código en Python (Versión Interactiva)

```
import math

def biseccion(f, a, b, tol=1e-6, max_iter=100):
    if f(a) * f(b) >= 0:
```

```
        print("    Error: f(a) y f(b) deben tener signos
            opuestos")
        return None

    print("\nIter\t a\t\t b\t\t m\t\t f(m)")
    for i in range(max_iter):
        m = (a + b) / 2
        fm = f(m)

        print(f"{i+1}\t {a:.6f}\t {b:.6f}\t {m:.6f}\t {fm:.6f
            }")

        if abs(fm) < tol:
            print("\ n   Convergencia alcanzada")
            return m

        if f(a) * fm < 0:
            b = m
        else:
            a = m

    print("\ n   Se alcanz  el n mero m ximo de
        iteraciones")
    return m

# ----- PROGRAMA PRINCIPAL -----

# Leer la funci n desde teclado
expr = input("Ingresa la funci n f(x): ")
# Ejemplo: x**3 - x - 2

# Crear la funci n a partir de la expresi n
f = lambda x: eval(expr, {"x": x, "math": math})

# Pedir los l mites del intervalo
a = float(input("Ingresa el extremo inferior a: "))
b = float(input("Ingresa el extremo superior b: "))

# Pedir tolerancia
tol = float(input("Ingresa la tolerancia (ej: 1e-6): "))
```

```
# Llamada al m todo
raiz = biseccion(f, a, b, tol)

print("\nLa ra z aproximada es:", raiz)
```

Conclusión

El método de bisección es uno de los procedimientos más simples y seguros para encontrar raíces de una función continua. Aunque no es el más rápido en comparación con otros métodos numéricos, garantiza convergencia siempre que la función cambie de signo en el intervalo inicial. Es especialmente útil como punto de partida en problemas de cálculo numérico y constituye una base fundamental para comprender técnicas más avanzadas de aproximación de soluciones.