Método de la Secante

Autor: Henry Ccoarite Dueñas
Curso: Programacion Numerica
Docente: (Fred Torres Cruz)
UNIVERSIDAD NACIONAL DEL ALTIPLANO - PUNO

1. Descripción del método

El **método de la secante** es un procedimiento numérico iterativo utilizado para **aproximar raíces reales** de funciones continuas, es decir, valores de x tales que $f(x) = 0$.

A diferencia del método de Newton-Raphson, este método **no requiere el cálculo analítico de la derivada**, sino que emplea una **aproximación de la derivada** mediante una recta secante que pasa por dos puntos consecutivos de la función.

Fórmula general

A partir de dos valores iniciales x_{n-1} y x_n, se calcula el siguiente valor mediante:

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

El proceso se repite hasta que el **error absoluto** entre dos iteraciones consecutivas sea menor que una tolerancia predefinida (tol), o hasta alcanzar un número máximo de iteraciones.

Ventajas

- No requiere el cálculo de derivadas, a diferencia del método de Newton-Raphson.
- Con frecuencia converge más rápido que el método de bisección.
- Es sencillo de implementar computacionalmente.

Desventajas

- No siempre garantiza la convergencia si los valores iniciales no son adecuados.
- Puede fallar si $f(x_n) - f(x_{n-1}) = 0$, generando una división por cero.

2. Código implementado en Python

A continuación se muestra el código del método de la secante en `Python`, debidamente comentado y estructurado.

Listing 1: Implementación del método de la secante en Python

```
import math

def secante(f, x0, x1, tol=1e-6, max_iter=100):

```

```
    print(f"{'Iter':<6} {'x_n-1':<15} {'x_n':<15} {'f(x_n)':<15} {'Error':<15}")
    print("-" * 65)

    x_prev = x0
    x_curr = x1

    print(f"{0:<6} {x0:<15.8f} {x1:<15.8f} {f(x1):<15.8f} {'-':<15}")

    for k in range(1, max_iter + 1):
        fx_prev = f(x_prev)
        fx_curr = f(x_curr)

        if fx_curr - fx_prev == 0:
            print("\nError: Division por cero. f(x_n) == f(x_n-1).")
            return None, k

        x_next = x_curr - fx_curr * (x_curr - x_prev) / (fx_curr - fx_prev)
        error = abs(x_next - x_curr)

        print(f"{k:<6} {x_curr:<15.8f} {x_next:<15.8f} {f(x_next):<15.8f} {error:<15.8e}")

        if error < tol:
            print("-" * 65)
            print(f"\nRa z encontrada: {x_next:.10f}")
            print(f"Iteraciones: {k}")
            return x_next, k

        x_prev = x_curr
        x_curr = x_next

    print("-" * 65)
    print(f"\nSe alcanz  el m ximo de iteraciones ({max_iter})")
    print(f" ltima  aproximaci n: {x_curr:.10f}")
    return x_curr, max_iter


if __name__ == "__main__":
    print("--- M TODO DE LA SECANTE ---")
    print("Ejemplo: Ra z de f(x) = x  - 4")
    print("Valores iniciales: x0 = 1, x1 = 3\n")

    def f_ejemplo(x):
        return x**2 - 4

    raiz, iteraciones = secante(f_ejemplo, 1, 3, tol=1e-6)
```

```
    print(f"Verificaci n: f({raiz:.10f}) = {f_ejemplo(raiz):.10e
      }")
```

3. Salida del programa (Output esperado)

El siguiente bloque muestra un ejemplo del resultado obtenido al ejecutar el programa anterior.

Salida del programa

```
--- MÉTODO DE LA SECANTE ---
Ejemplo: Raíz de f(x) = x² - 4
Valores iniciales: x0 = 1, x1 = 3

Iter   x_n-1          x_n             f(x_n)          Error
-----------------------------------------------------------------
0      1.00000000     3.00000000      5.00000000      -
1      3.00000000     2.33333333      1.44444444      6.66666667e-01
2      2.33333333     2.06293706      0.25626327      2.70396270e-01
3      2.06293706     2.00026092      0.00104374      6.26761392e-02
4      2.00026092     2.00000003      0.00000011      2.60895272e-04
-----------------------------------------------------------------


Raíz encontrada: 2.0000000325
Iteraciones: 4
Verificación: f(2.0000000325) = 0.0000000000e+00
```

4. Descripción del funcionamiento del código

1. Se importa la librería `math` para permitir el uso de funciones matemáticas.
2. Se define la función `secante()`, que recibe como parámetros la función f, los valores iniciales x_0 y x_1, la tolerancia y el número máximo de iteraciones.
3. Dentro del bucle, se calcula x_{n+1} usando la fórmula de la secante, y se evalúa el error absoluto entre dos iteraciones consecutivas.
4. Si el error es menor que la tolerancia, el programa detiene la iteración e imprime la raíz aproximada.
5. En caso de alcanzar el número máximo de iteraciones sin converger, muestra la última aproximación obtenida.
6. Finalmente, en el bloque principal se aplica el método a la función $f(x) = x^2 - 4$, mostrando la raíz encontrada y su verificación.

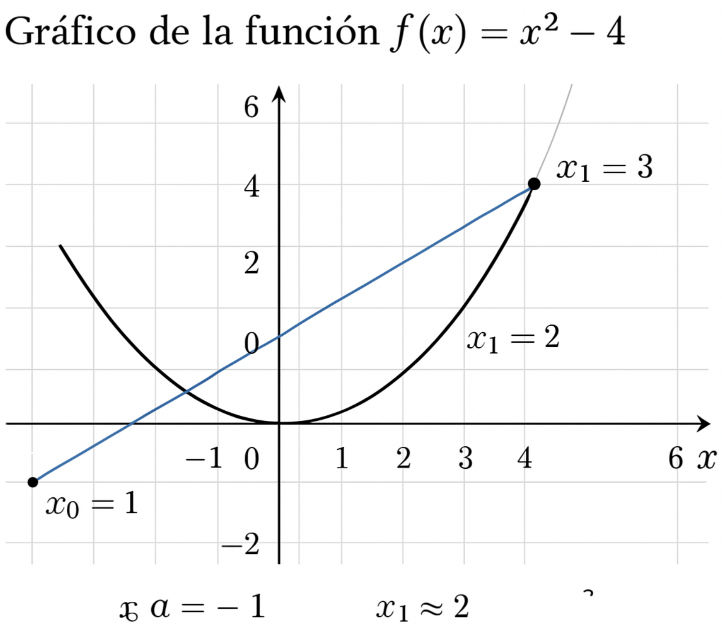

Figura 1: Grafico

5. Conclusión

El método de la secante es una herramienta eficaz para encontrar raíces de ecuaciones no lineales sin necesidad de derivar la función. Su implementación en Python demuestra que es un método sencillo, rápido y versátil, aunque requiere una adecuada selección de valores iniciales para asegurar la convergencia hacia la raíz deseada.