Título del proyecto:

“Interfaz métodos abiertos final.”

Autor:

Aldo Antonio Sánchez Pérez.

Institución:

Tecnológico Nacional De México, Campus Tuxtla Gutiérrez.

Fecha:

04/04/2025

Resumen.

Este proyecto tiene como objetivo implementar una interfaz gráfica interactiva para resolver ecuaciones no lineales usando los métodos de Secante, Bisección y Newton-Raphson. Utilizando Python y Tkinter, se desarrolla una herramienta visual donde el usuario puede ingresar funciones, seleccionar el método deseado y obtener la raíz de la ecuación, junto con su tabla de iteraciones y gráfica.

Introducción.

Los métodos numéricos son fundamentales para resolver ecuaciones no lineales cuando no es posible obtener soluciones analíticas. Los métodos de Secante y Newton-Raphson son algoritmos de aproximación que convergen rápidamente bajo ciertas condiciones.

El uso de interfaces gráficas permite facilitar la interacción del usuario con estos algoritmos, eliminando la necesidad de escribir código cada vez que se desea resolver una ecuación.

Objetivo general: Crear una aplicación gráfica en Python que implemente los métodos abiertos y brinde una experiencia amigable al usuario.

Fundamento Teórico.

Método de la Secante:

- **Ecuación base:** Se basa en una aproximación lineal a partir de dos puntos.

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

- **Condiciones:** Requiere dos valores iniciales cercanos a la raíz.
- **Convergencia:** Superlineal si los valores iniciales están bien elegidos.

Método de Newton-Raphson:

- **Fórmula:**

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

- **Condiciones:** Necesita derivar la función.

- **Riesgos:** Puede divergir si la derivada se acerca a cero o si el valor inicial está lejos de la raíz.

Diseño de la Interfaz.

Herramientas:

- Python
- Tkinter
- SymPy (para operaciones simbólicas)
- Matplotlib (para graficación)

Estructura General:

- Entradas para función, valores iniciales, umbral y máximo de iteraciones.
- Selección de método.
- Botón para calcular.
- Tabla de iteraciones y gráfica del resultado.

Esquema de interacción:

- El usuario elige el método y los datos.
- Se valida la entrada.
- Se ejecuta el algoritmo y se muestran resultados.

Desarrollo del Código.

Validación de Datos:

- Se verifica que la función esté bien escrita.
- Se comprueba que los valores iniciales sean válidos según el método.

Secante:

- Se implementa la fórmula con iteración y control de tolerancia.
- Se lanza advertencia si los valores no cumplen condiciones.

Newton-Raphson:

- Se calcula la derivada usando SymPy.
- Se implementa la fórmula con verificación de divergencia.

Salidas:

- Tabla con iteraciones:

$$x_i, f(x_i)$$

- Gráfica con la raíz marcada.

Ejemplos y Pruebas.

Casos de prueba con funciones conocidas:

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

- Raíz esperada:
  $$\sqrt{2} \approx 1.4142$$

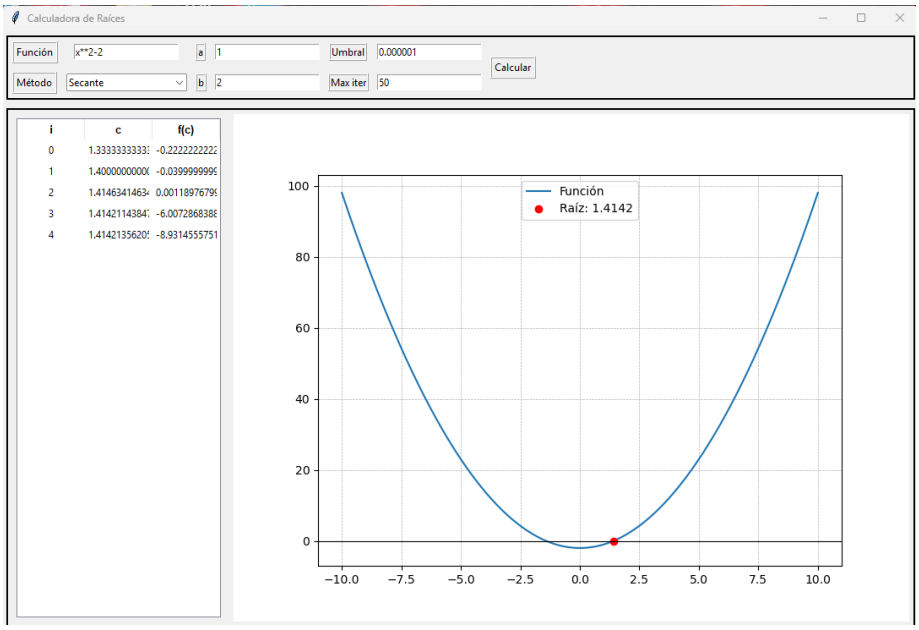
Comparativa de métodos con mismos datos iniciales:

- Se probó con la función anterior, con valores iniciales 1 y 2 para la secante y para Newton-Raphson solo un valor que es 2:

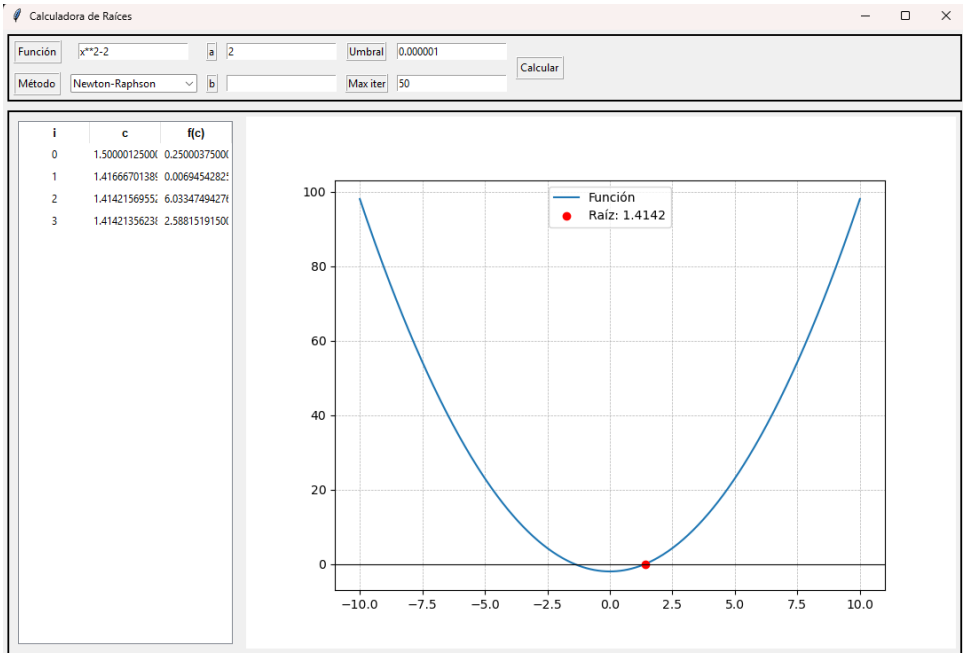
- **Secante**: convergió en 4 iteraciones.
- **Newton-Raphson**: convergió en 3 iteraciones.

Capturas de pantalla:

- Interfaz ejecutando el método de Secante.



- Interfaz ejecutando el método de Newton-Raphson.



Resultados y Discusión.

- Ambos métodos (Secante y Newton-Raphson) convergen rápidamente si los datos iniciales son adecuados.
- La interfaz facilita la interacción y comprensión de los métodos numéricos.

Conclusiones.

- Se logró implementar correctamente los métodos abiertos.
- La interfaz es funcional, clara y gráfica.
- Se identificó que el método Newton-Raphson puede fallar si la derivada es cero.

Referencias.

- Chapra, Steven. Métodos numéricos para ingenieros.
- Documentación oficial de Python, SymPy y Tkinter.

Anexos.

- Código fuente.
- Capturas.
- Manual de usuario (si aplica).