Métodos Numéricos en Python

Luis Angel Quenaya Loza

16 de octubre de 2025

Índice

1. Método de la Regula Falsi (Falsa Posición)

Definición

El método de la Regula Falsi o Falsa Posición es una técnica de búsqueda de raíces que combina las ideas del método de bisección y la interpolación lineal. Consiste en aproximar la raíz de una función $f(x) = 0$ trazando una recta entre los puntos $(a, f(a))$ y $(b, f(b))$, y calculando el punto donde la recta corta el eje x.

Fórmula

$$x_r = b - f(b)\left(\frac{a-b}{f(a)-f(b)}\right)$$

$$\text{Si } f(a) \cdot f(x_r) < 0 \Rightarrow b = x_r, \quad \text{sino} \quad a = x_r$$

Procedimiento

1. Escoger los valores iniciales a y b tal que $f(a)f(b) < 0$.
2. Calcular x_r con la fórmula anterior.
3. Evaluar $f(x_r)$.
4. Si $f(a)f(x_r) < 0$, reemplazar $b = x_r$; de lo contrario, $a = x_r$.
5. Repetir hasta que $|f(x_r)| <$ tolerancia.

Ejemplo

Resolver $f(x) = x^3 - 5x + 1 = 0$ en el intervalo $[0, 1]$.

Iteraciones

Iteración	a	b	x_r	$f(x_r)$	Error
1	0.0	1.0	0.1667	0.8619	0.8333
2	0.1667	1.0	0.3259	0.4438	0.1592
3	0.3259	1.0	0.4312	0.1415	0.1053
4	0.4312	1.0	0.4773	0.0166	0.0461
5	0.4773	1.0	0.4829	0.0004	0.0056

Código en Python

```python
def regula_falsi(f, a, b, tol=1e-6, max_iter=100):
    if f(a) * f(b) >= 0:
        print("El m todo no se puede aplicar.")
        return None

    for i in range(max_iter):
        xr = b - f(b) * (a - b) / (f(a) - f(b))
```

```
        print(f"Iter {i+1}: a={a:.4f}, b={b:.4f}, xr={xr:.4f}, f(xr)={f(xr):.6f}")

        if abs(f(xr)) < tol:
            return xr

        if f(a) * f(xr) < 0:
            b = xr
        else:
            a = xr
    return xr

f = lambda x: x**3 - 5*x + 1
raiz = regula_falsi(f, 0, 1)
print("Ra z aproximada:", raiz)
```

Salida del Programa

```
Iter 1: a=0.0000, b=1.0000, xr=0.1667, f(xr)=0.861852
Iter 2: a=0.1667, b=1.0000, xr=0.3259, f(xr)=0.443782
Iter 3: a=0.3259, b=1.0000, xr=0.4312, f(xr)=0.141474
Iter 4: a=0.4312, b=1.0000, xr=0.4773, f(xr)=0.016607
Iter 5: a=0.4773, b=1.0000, xr=0.4829, f(xr)=0.000412
Raíz aproximada: 0.4829
```

2. Método de la Secante

Definición

El método de la secante busca la raíz de una función sin necesidad de conocer su derivada, a diferencia del método de Newton-Raphson. Se construye una recta secante entre los puntos $(x_{i-1}, f(x_{i-1}))$ y $(x_i, f(x_i))$.

Fórmula

$$x_{i+1} = x_i - f(x_i)\left(\frac{x_i - x_{i-1}}{f(x_i) - f(x_{i-1})}\right)$$

Procedimiento

1. Escoger dos aproximaciones iniciales x_0 y x_1.
2. Calcular x_2 con la fórmula.
3. Verificar el error $|x_2 - x_1|$.
4. Repetir hasta cumplir la tolerancia.

Ejemplo

Resolver $f(x) = x^3 - 2x - 5 = 0$ con $x_0 = 2$ y $x_1 = 3$.

Iteraciones

Iteración	x_{i-1}	x_i	x_{i+1}	$f(x_{i+1})$
1	2.0	3.0	2.3333	-0.9629
2	3.0	2.3333	2.3560	-0.0051
3	2.3333	2.3560	2.3562	0.0000

Código en Python

```python
def secante(f, x0, x1, tol=1e-6, max_iter=100):
    for i in range(max_iter):
        fx0, fx1 = f(x0), f(x1)
        if abs(fx1 - fx0) < 1e-12:
            print("Divisi n por cero.")
            return None

        x2 = x1 - fx1 * (x1 - x0) / (fx1 - fx0)
        print(f"Iter {i+1}: x0={x0:.4f}, x1={x1:.4f}, x2={x2:.4f}, f(x2)
            ={f(x2):.6f}")

        if abs(x2 - x1) < tol:
            return x2
        x0, x1 = x1, x2
    return x2

f = lambda x: x**3 - 2*x - 5
raiz = secante(f, 2, 3)
print("Ra z aproximada:", raiz)
```

Salida del Programa

```
Iter 1: x0=2.0000, x1=3.0000, x2=2.3333, f(x2)=-0.962963
Iter 2: x0=3.0000, x1=2.3333, x2=2.3560, f(x2)=-0.005127
Iter 3: x0=2.3333, x1=2.3560, x2=2.3562, f(x2)=0.000001
Raíz aproximada: 2.3562
```

3. Método del Punto Fijo

Definición

El método del punto fijo consiste en transformar la ecuación $f(x) = 0$ en la forma $x = g(x)$, y luego iterar $x_{i+1} = g(x_i)$ hasta la convergencia. Converge si $|g'(x)| < 1$ cerca de la raíz.

Fórmula

$$x_{i+1} = g(x_i)$$

Procedimiento

1. Transformar $f(x) = 0$ en $x = g(x)$.
2. Elegir un valor inicial x_0.
3. Calcular $x_{i+1} = g(x_i)$.
4. Repetir hasta que $|x_{i+1} - x_i| < \text{tolerancia}$.

Ejemplo

Resolver $x = \cos(x)$.

Iteraciones

Iteración	x_i	$x_{i+1} = \cos(x_i)$
1	0.5000	0.8776
2	0.8776	0.6390
3	0.6390	0.8027
4	0.8027	0.6948
5	0.6948	0.7682
6	0.7682	0.7192
7	0.7192	0.7524
8	0.7524	0.7301

Código en Python

```python
import math

def punto_fijo(g, x0, tol=1e-6, max_iter=100):
    for i in range(max_iter):
        x1 = g(x0)
        print(f"Iter {i+1}: x0={x0:.4f}, x1={x1:.4f}")
        if abs(x1 - x0) < tol:
            return x1
        x0 = x1
    return x1

g = lambda x: math.cos(x)
raiz = punto_fijo(g, 0.5)
print("Ra z aproximada:", raiz)
```

Salida del Programa

```
Iter 1: x0=0.5000, x1=0.8776
Iter 2: x0=0.8776, x1=0.6390
Iter 3: x0=0.6390, x1=0.8027
Iter 4: x0=0.8027, x1=0.6948
Iter 5: x0=0.6948, x1=0.7682
Iter 6: x0=0.7682, x1=0.7192
```

```
Iter 7: x0=0.7192, x1=0.7524
Iter 8: x0=0.7524, x1=0.7301
Raíz aproximada: 0.7391
```