

# LAPORAN PRAKTIKUM 1

## Kelompok 16

Nama Anggota Kelompok :

- Lucky Himawan Prasetya (5025241147)
- Muh. Aqil Alqadri Syahid (5025241161)
- Hosea Felix Sanjaya (5025241177)

Link Github Terkait :

<https://github.com/arkananta47/Praktikum-Komnum/blob/main/praktikum1.py>

Anda sudah mengerti algoritma pemrosesan metode Regula Falsi, dan anda sudah memahami cara kerjanya. Sekarang anda tinggal mengimplementasikan algoritma tersebut menjadi sebuah program komputer metode Regula Falsi (yang dapat menampilkan proses iteratif numerik, lengkap dengan grafik fungsinya).

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

```
# BOLEH DIUBAH YA
```

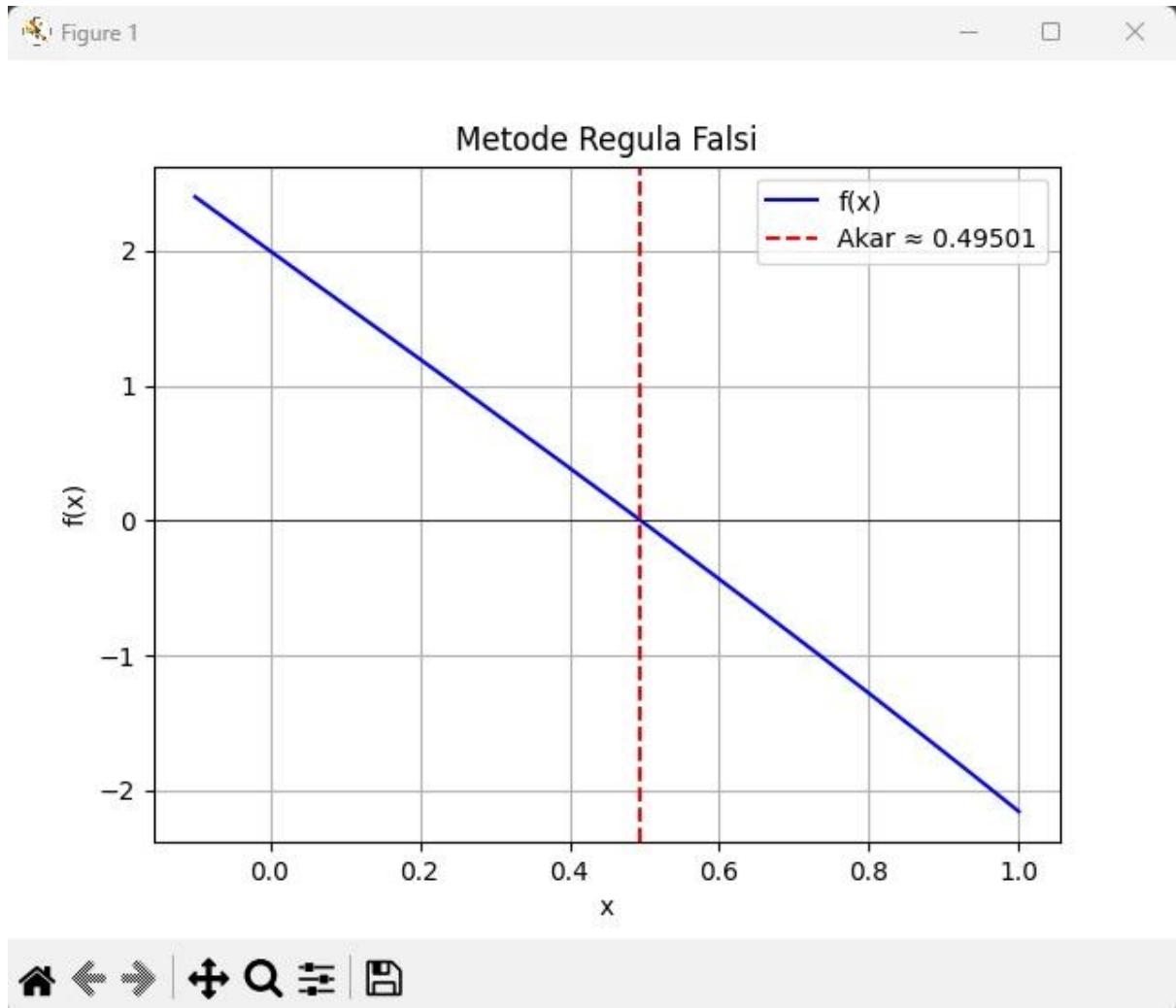
```
def f(x):  
    return math.cos(x) - 3*x  
  
x1 = 0.3  
  
x2 = 0.4
```

```
# Contoh Soal 1:
```

```
# return math.sin(x) - 5*x + 2  
  
# x1 = 0.4  
  
# x2 = 0.5
```

Iterasi	x1	x2	x3	f(x3)
1	0.400000	0.500000	0.494982	0.000107
2	0.494982	0.500000	0.495008	0.000000

Akar ditemukan pada x = 0.495008



# Contoh Soal 2:

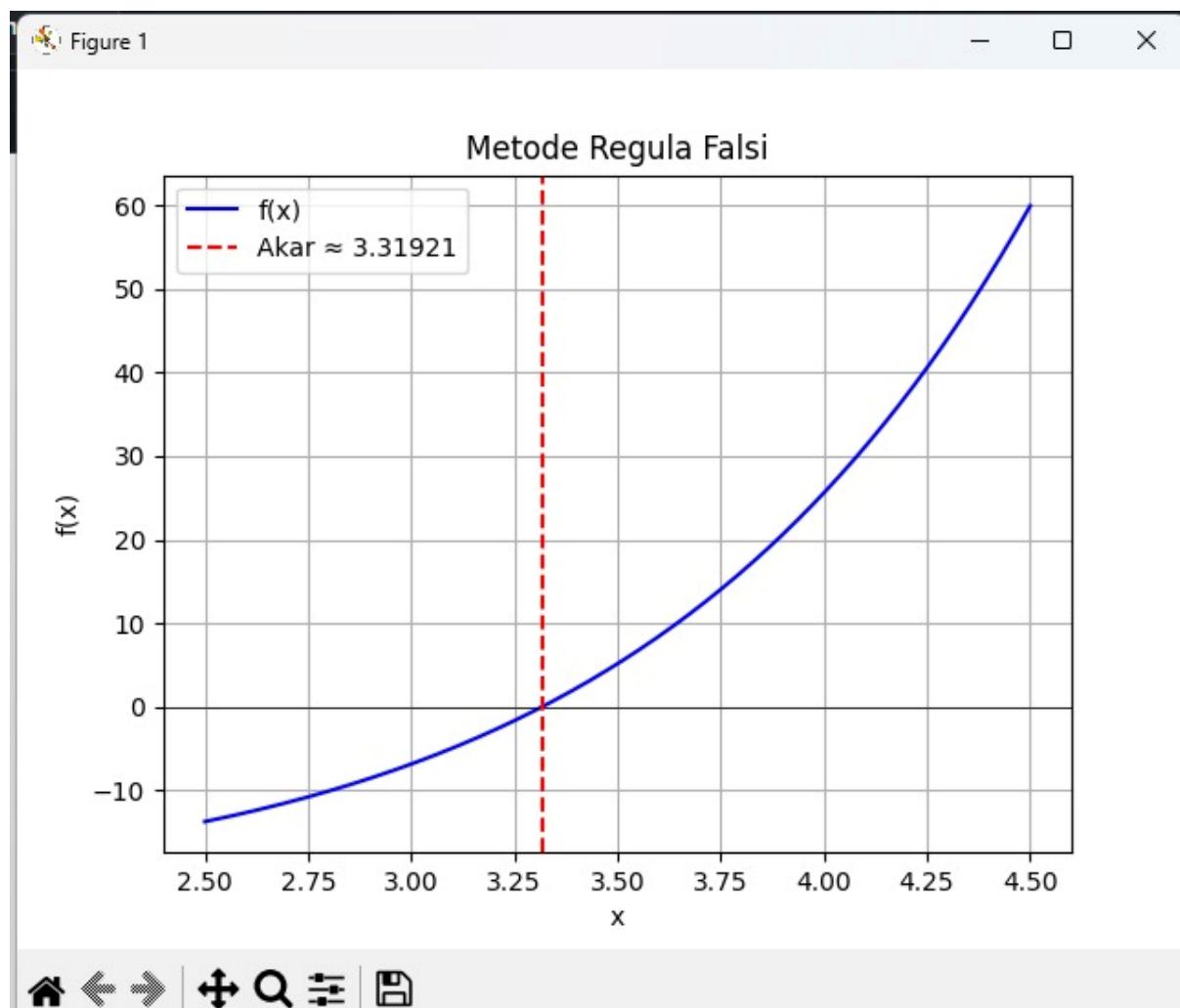
```
# return math.exp(x) - 2*x - 21
```

```
# x1 = 3
```

```
# x2 = 4
```

Iterasi	x1	x2	x3	f(x3)
1	3.000000	4.000000	3.212670	-2.580001
2	3.212670	4.000000	3.284758	-0.866985
3	3.284758	4.000000	3.308189	-0.280793
4	3.308189	4.000000	3.315696	-0.089844
5	3.315696	4.000000	3.318089	-0.028635
6	3.318089	4.000000	3.318851	-0.009115
7	3.318851	4.000000	3.319093	-0.002900
8	3.319093	4.000000	3.319171	-0.000923
9	3.319171	4.000000	3.319195	-0.000294
10	3.319195	4.000000	3.319203	-0.000093
11	3.319203	4.000000	3.319205	-0.000030
12	3.319205	4.000000	3.319206	-0.000009

Akar ditemukan pada  $x = 3.319206$



### # Contoh Soal 3:

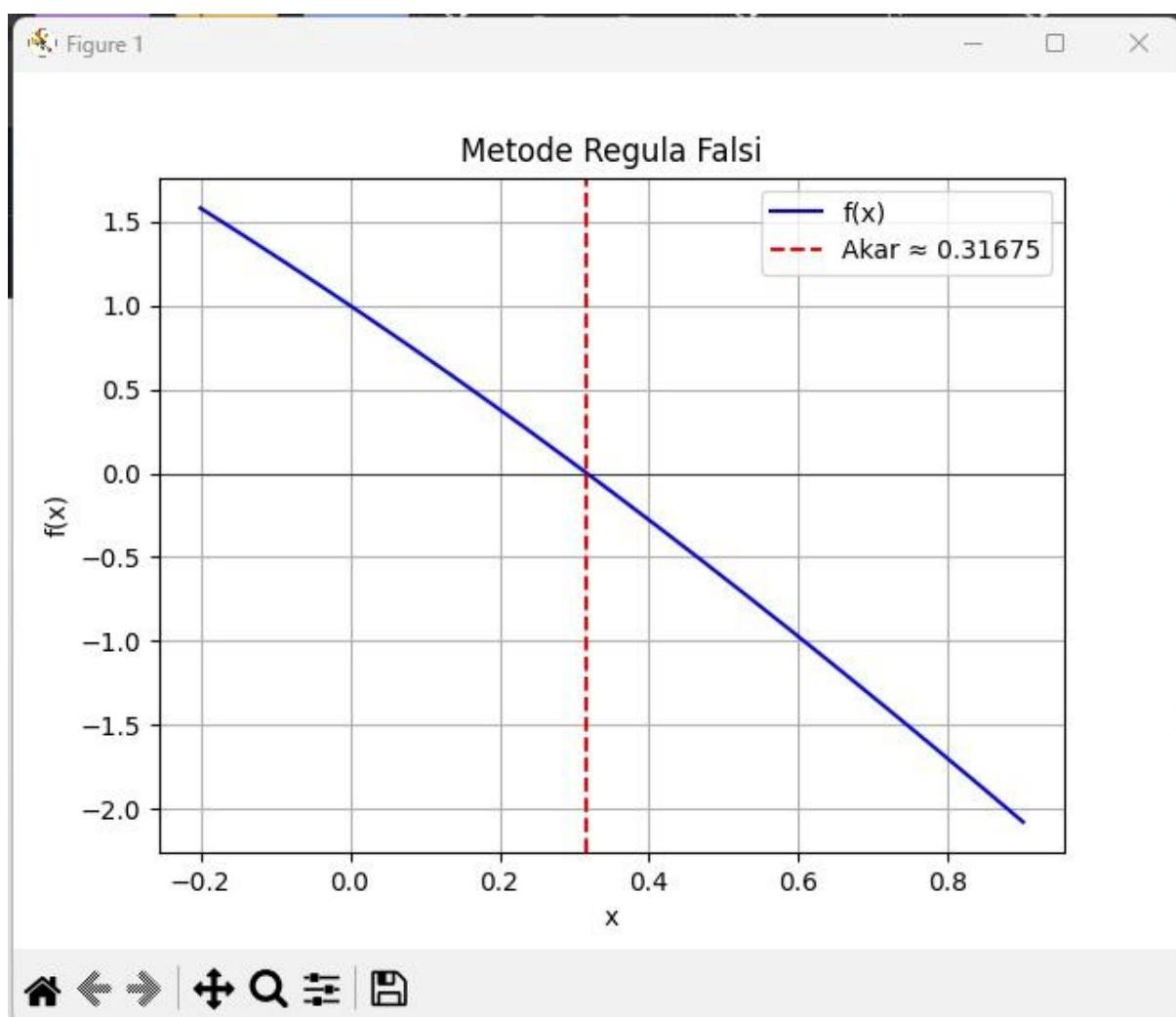
```
# return math.cos(x) - 3*x
```

# x1 = 0.3

# x2 = 0.4

Iterasi	x1	x2	x3	f(x3)
1	0.300000	0.400000	0.316554	0.000651
2	0.316554	0.400000	0.316749	0.000008

Akar ditemukan pada  $x = 0.316749$



## # Contoh Soal 4:

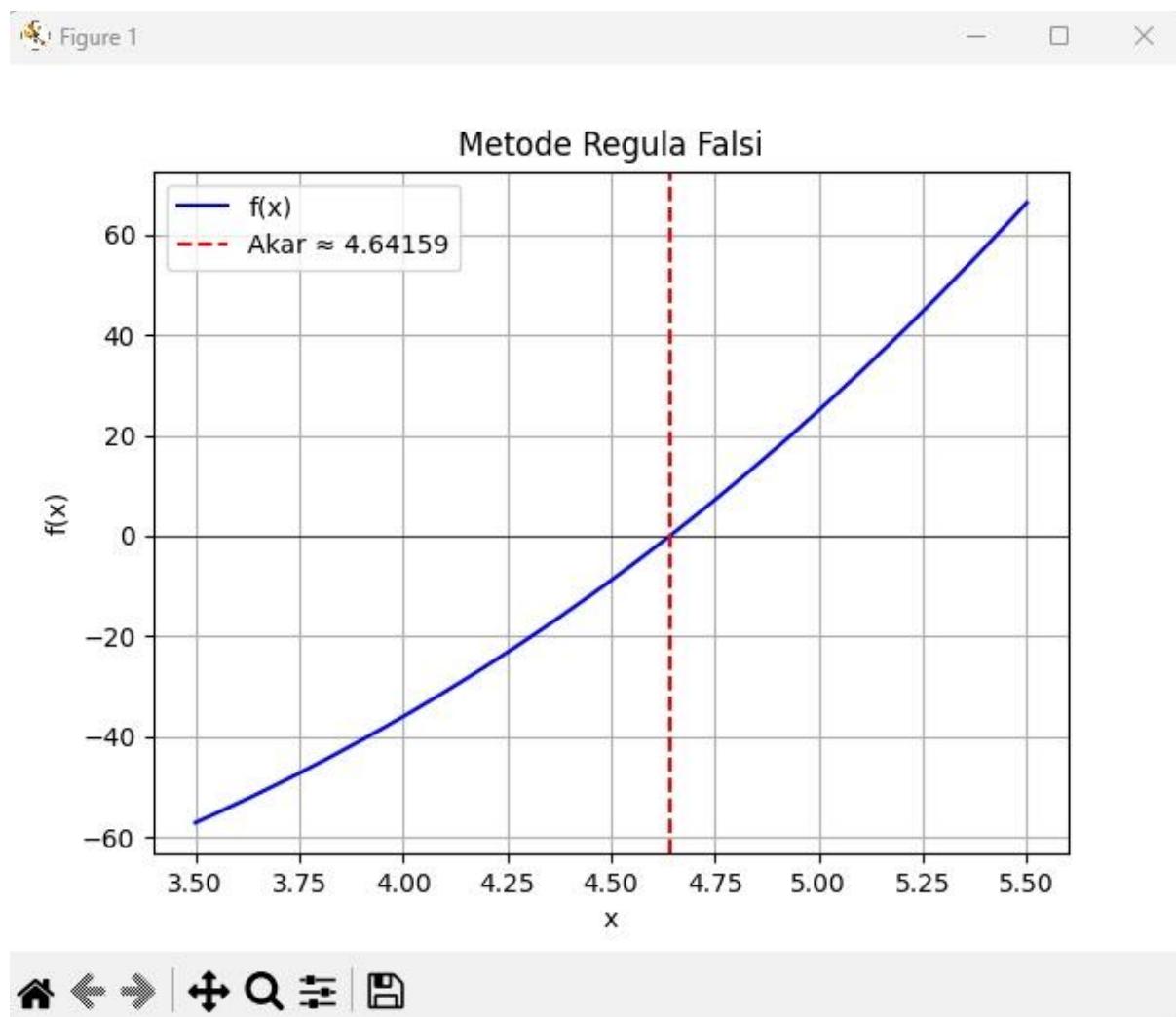
```
# return x**3 - 100
```

# x1 = 4

# x2 = 5

Iterasi	x1	x2	x3	f(x3)
1	4.000000	5.000000	4.590164	-3.287059
2	4.590164	5.000000	4.637788	-0.245434
3	4.637788	5.000000	4.641310	-0.018036
4	4.641310	5.000000	4.641568	-0.001324
5	4.641568	5.000000	4.641587	-0.000097
6	4.641587	5.000000	4.641589	-0.000007

Akar ditemukan pada  $x = 4.641589$



tol = 1e-5

max\_iter = 100

```
def regula_falsi(f, x1, x2, tol=1e-5, max_iter=100):
    if f(x1) * f(x2) >= 0:
        print(f"Metode gagal: f({x1}) = {f(x1):.6f}, f({x2}) = {f(x2):.6f}")
    else:
        for i in range(max_iter):
            x3 = x1 - (f(x1) * (x2 - x1)) / (f(x2) - f(x1))
            if abs(f(x3)) < tol:
                return x3
            if f(x1) * f(x3) <= 0:
                x2 = x3
            else:
                x1 = x3
        return x3
```

```

print("f(x1) dan f(x2) harus memiliki tanda yang berbeda.")

return None

print("{}{:<8}{:<12}{:<12}{:<12}{:<12}".format("Iterasi", "x1", "x2", "x3", "f(x3)"))

for i in range(max_iter):

    f1 = f(x1)

    f2 = f(x2)

    x3 = x2 - f2 * (x1 - x2) / (f1 - f2)

    f3 = f(x3)

    print("{}{:<8}{:<12.6f}{:<12.6f}{:<12.6f}{:<12.6f}".format(i + 1, x1, x2, x3, f3))

    if abs(f3) < tol:

        return x3

    if f1 * f3 < 0:

        x2 = x3

    else:

        x1 = x3

return x3

akar = regula_falsi(f, x1, x2, tol, max_iter)

if akar is not None:

    print("\nAkar ditemukan pada x = {:.6f}".format(akar))

x_vals = np.linspace(x1 - 0.5, x2 + 0.5, 400)

y_vals = [f(x) for x in x_vals]

plt.plot(x_vals, y_vals, label="f(x)", color='blue')

plt.axhline(0, color='black', linewidth=0.5)

plt.axvline(akar, color='red', linestyle='--', label=f"Akar ≈ {akar:.5f}")

```

```

plt.title("Metode Regula Falsi")

plt.xlabel("x")

plt.ylabel("f(x)")

plt.grid(True)

plt.legend()

plt.show()

```

### Ide Pendekatan : **Regula Falsi**

Kode diatas menggunakan Metode Regula Falsi (False Position Method) untuk mencari akar dari suatu fungsi non-linear  $f(x)$ , yaitu nilai  $x$  yang memenuhi  $f(x) = 0$ . Tujuannya untuk mencari akar dari fungsi  $f(x)$  menggunakan metode numerik karena tidak semua fungsi bisa diselesaikan secara analitik (langsung dengan rumus).

Metode Regula Falsi adalah pendekatan numerik untuk menemukan akar fungsi dalam selang  $[x_1, x_2]$  dengan syarat:

-  $f(x_1) * f(x_2) < 0 \rightarrow$  artinya tanda dari fungsi di kedua titik berbeda  $\rightarrow$  ada akar di antaranya (berdasarkan Teorema Bolzano).

Langkah-langkah metode Regula Falsi:

#### **1. Hitung titik tengah $x_3$ menggunakan rumus interpolasi linier:**

$$x_3 = x_2 - f(x_2) * (x_1 - x_2) / (f(x_1) - f(x_2))$$

#### **2. Evaluasi $f(x_3)$ :**

- Jika  $|f(x_3)| <$  toleransi, maka akar ditemukan.
- Jika  $f(x_1) * f(x_3) < 0 \rightarrow$  akar berada di antara  $x_1$  dan  $x_3 \rightarrow$  set  $x_2 = x_3$ .
- Jika  $f(x_2) * f(x_3) < 0 \rightarrow$  akar berada di antara  $x_3$  dan  $x_2 \rightarrow$  set  $x_1 = x_3$ .

#### **3. Ulangi proses hingga akar ditemukan atau jumlah iterasi maksimum tercapai.**

Penjelasan Kode:

1. Fungsi  $f(x)$  didefinisikan terlebih dahulu (dapat diubah).
2. Validasi awal memastikan bahwa  $f(x_1)$  dan  $f(x_2)$  memiliki tanda berbeda.
3. Iterasi dilakukan menggunakan rumus Regula Falsi.
4. Jika akar ditemukan, hasil divisualisasikan menggunakan matplotlib.

Kelebihan dan Kekurangan Metode Regula Falsi:

Kelebihan :

- Lebih cepat dari metode biseksi dalam banyak kasus.
- Mempersempit interval sambil mempertahankan nilai yang lebih akurat.

Kekurangan :

- Jika fungsi mendekati linier atau mendatar di salah satu sisi, metode bisa konvergen sangat lambat.

Contoh Soal Alternatif:

- $\text{math.sin}(x) - 5*x + 2$
- $\text{math.exp}(x) - 2*x - 21$

Ringkasan:

Metode Regula Falsi bekerja dengan menggunakan interpolasi linier antara dua titik yang memiliki tanda fungsi berbeda, lalu memperbaiki interval berdasarkan nilai fungsi titik tengah. Ini adalah metode numerik klasik untuk pencarian akar fungsi non-linear.