Metodo de bisseção e new-rapson

import numpy as np

import matplotlib.pyplot as plt

def f(x):

return np.cos(x)\*\*2 - np.sin(x)

def df(x):

return -2\*np.cos(x)\*np.sin(x) - np.cos(x)

def newton\_raphson(x0, tol, max\_iter):

x = x0

for i in range(max\_iter):

x\_new = x - f(x)/df(x)

if abs(x\_new - x) < tol:

return x\_new

x = x\_new

print("Método de Newton-Raphson não convergiu")

return None

def bissecao(a, b, tol, max\_iter):

for i in range(max\_iter):

c = (a + b)/2

if f(c) == 0 or (b - a)/2 < tol:

return c

elif f(a)\*f(c) < 0:

b = c

else:

a = c

print("Método da Bisseção não convergiu")

return None

# Parâmetros

a = 1.5

b = 4

tol = 1e-6

max\_iter = 100

x0 = 2 # Chute inicial para Newton-Raphson

# Encontrar as raízes

raiz\_newton = newton\_raphson(x0, tol, max\_iter)

raiz\_bissecao = bissecao(a, b, tol, max\_iter)

print("Raiz encontrada pelo método de Newton-Raphson:", raiz\_newton)

print("Raiz encontrada pelo método da Bisseção:", raiz\_bissecao)

# Plotar o gráfico

x = np.linspace(1.5, 4, 100)

plt.plot(x, f(x))

plt.grid(True)

plt.xlabel('x')

plt.ylabel('f(x)')

plt.title('Gráfico da função e raízes encontradas')

plt.show()

Achar as derivadas

import sympy as sp

# Definindo a variável simbólica

x = sp.symbols('x')

# Definindo a função

f = x\*\*3 - 5\*x + sp.sin(x)

# Calculando a derivada da função

derivada\_f = sp.diff(f, x)

# Exibindo a derivada

print(derivada\_f)