Conjunto de ejercicios 3

Boris Garcés

Tabla de Contenidos

Conjunto de ejercicios

1. Use el método de bisección para encontrar soluciones precisas dentro de 10^{-2} para $x^3 - 7x^2 + 14x - 6 = 0$ en cada intervalo.

a) $[0, 1]$

```
def metodo_biseccion(funcion, inicio, fin, tolerancia=1e-5):
    if funcion(inicio) * funcion(fin) >= 0:
        print("La función no cambia de signo en el intervalo dado. Puede que no haya raíces e
        return None
    while abs(fin - inicio) >= tolerancia:
        punto_medio = (inicio + fin) / 2
        valor_medio = funcion(punto_medio)
        if valor_medio == 0:
            return punto_medio
        if funcion(inicio) * valor_medio < 0:
            fin = punto_medio
        else:
            inicio = punto_medio
    return (inicio + fin) / 2
def F(x):
    return x**3 - 7*x**2 + 14*x - 6
```

```
intervalo_inicio, intervalo_fin = 0, 1
raiz = metodo_biseccion(F, intervalo_inicio, intervalo_fin)
print(f"La raíz en el intervalo [{intervalo_inicio}, {intervalo_fin}] es: {raiz}")
```

```
La raíz en el intervalo [0, 1] es: 0.5857887268066406
```

b)$[1, 3.2]$

```
inicio_intervalo, fin_intervalo = 1, 3.2
raiz = metodo_biseccion(F, inicio_intervalo, fin_intervalo)
print(f"La raíz en el intervalo [{inicio_intervalo}, {fin_intervalo}] es: {raiz}")
```

```
La raíz en el intervalo [1, 3.2] es: 2.9999980926513676
```

c)$[3.2, 4]$

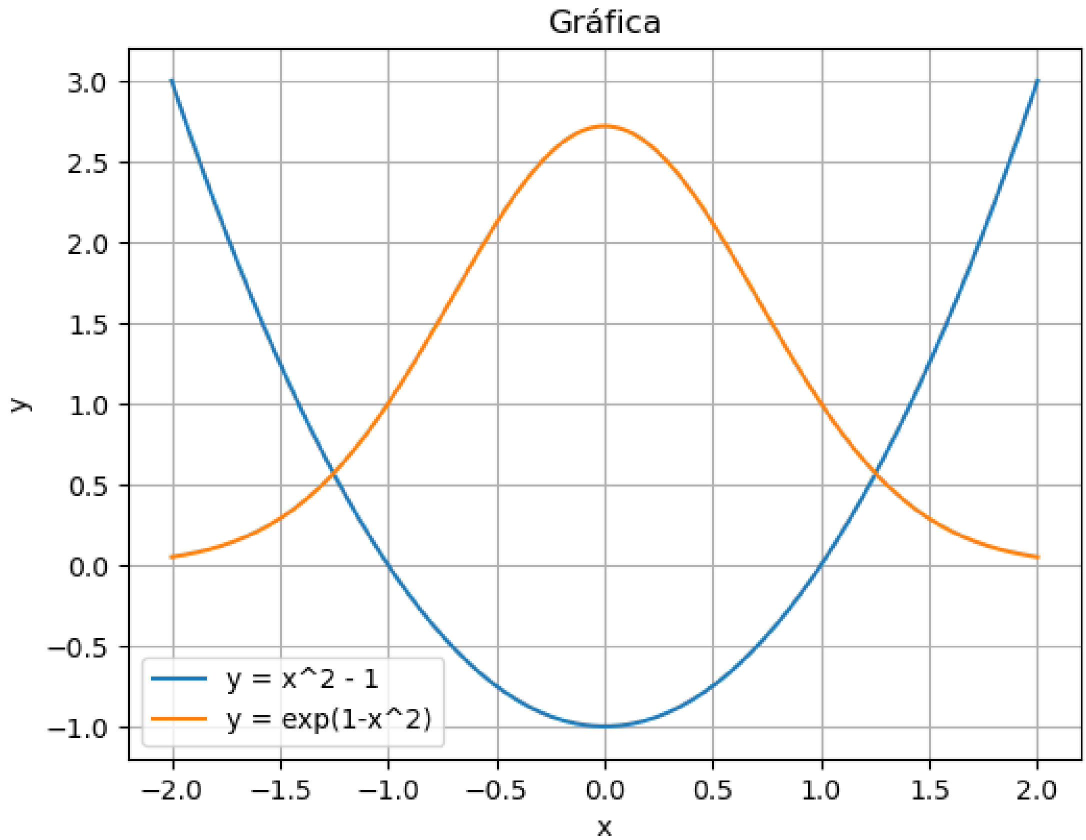
```
inicio_intervalo, fin_intervalo = 3, 2.4
raiz = metodo_biseccion(F, inicio_intervalo, fin_intervalo)
print(f"La raíz en el intervalo [{inicio_intervalo}, {fin_intervalo}] es: {raiz}")
```

```
La función no cambia de signo en el intervalo dado. Puede que no haya raíces o haya múltiples
La raíz en el intervalo [3, 2.4] es: None
```

4.

a)Dibuje las gráficas para $y = x$ y $y = \tan x$

```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(-2, 2, 400)
y1, y2 = x**2 - 1, np.exp(1-x**2)
plt.plot(x, y1, label="y = x^2 - 1")
plt.plot(x, y2, label="y = exp(1-x^2)")
plt.xlabel("x")
plt.ylabel("y")
plt.title("Gráfica")
plt.legend()
plt.grid(True)
plt.show()
```



b) Use el método de la bisección para encontrar una aproximación dentro de 10^-3 para un valor en $[-2, 0]$ con $x^2 - 1 = e^{1-x^2}$

```
import numpy as np
def bisection_method(f, a, b, tol=1e-5):
    while abs(b-a) > tol:
        c = (a+b)/2
        if f(c) == 0.0:
            return c
        elif f(a)*f(c) < 0:
            b = c
        else:
            a = c
    return (a+b)/2
def F1(x):
    return x**2 - 1 - np.exp(1-x**2)
X0, XF = -2, 0
```

```
root = bisection_method(F1, X0, XF)
print(f"La raíz para la función es: {root}")
```

```
La raíz para la función es: -1.2518577575683594
```

Ejercicios Aplicados

1. Un abrevadero de longitud tiene una sección transversal en forma de semicírculo con radio . (Consulte la figura adjunta.) Cuando se llena con agua hasta una distancia a partir de la parte superior, el volumen d ag es $V = L[0.5\pi r^2 - arcsen(\frac{h}{r} - h(r^2 - h^2)^{\frac{1}{2}})]$es

```
import numpy as np

def bisection_method(f, a, b, tol=1e-5):
    while abs(b-a) > tol:
        c = (a+b)/2
        if f(c) == 0.0:
            return c
        elif f(a)*f(c) < 0:
            b = c
        else:
            a = c
    return (a+b)/2

def V(h, L=10, r=1):
    return L * (0.5 * np.pi * r**2 - r**2 * np.arcsin(h/r) - h * np.sqrt(r**2 - h**2)) - 12.4

a, b = 0, 1  # Initial interval
root = bisection_method(V, a, b, tol=0.01)
print(f"La profundidad del agua es: {root} cm")
```

```
La profundidad del agua es: 0.16796875 cm
```

2. Un objeto que cae verticalmente a través del aire está sujeto a una resistencia viscosa, así como a la fuerza de gravedad. Suponga que un objeto con masa m cae desde una altura s_0 y que la altura del objeto después de t segundos es

$$s(t) = s_0 - \frac{mg}{k}t + \frac{m^2 g}{k^2}(1 - e^{-kt/m}),$$

donde $g = 9.81m/s^2$ y k representa el coeficiente de la resistencia del aire en Ns/m. Suponga $s_0 = 300m$, $m = 0.25kg$ y $k = 0.1Ns/m$. Encuentre dentro de $0.01 segundos$, el tiempo que tarda un cuarto de kg en golpear el piso.

```
import numpy as np
altura_inicial = 300
masa = 0.25
gravedad = 9.81
constante_resistencia = 0.1
def altura(t):
    return (
        altura_inicial
        - (masa * gravedad / constante_resistencia) * t
        + (masa**2 * gravedad / constante_resistencia**2) * (1 - np.exp(-constante_resistenci
    )
limite_inferior, limite_superior = 0, 100
tolerancia = 0.01
tiempo_impacto = metodo_biseccion(altura, limite_inferior, limite_superior, tolerancia)
print(f"El tiempo que tarda en golpear el piso es aproximadamente: {tiempo_impacto} segundos'
```

```
El tiempo que tarda en golpear el piso es aproximadamente: None segundos
```

Ejercicios teóricos

1. Use el teorrema 2.1 para encontrar una cota para el número de iteraciones necesarias para lograr una aproximación con precisión de 10^{-4} para la solución de $x^3 - x - 1 = 0$ que se encuentra dentro del intervalo $[1, 2]$.Encuentre una aproximación para la raíz con este grado de precisión

```
import numpy as np
def funcion(x):
    return x**3 - x - 1
limite_inferior, limite_superior = 1, 2
tolerancia = 1e-4
iteraciones_minimas = int(np.ceil((np.log(limite_superior - limite_inferior) - np.log(tolerar
print(f"Iteraciones mínimas necesarias según el Teorema 2.1: {iteraciones_minimas}")
aproximacion_raiz = metodo_biseccion(funcion, limite_inferior, limite_superior)
print(f"La aproximación de la raíz es: {aproximacion_raiz}")
```

```
Iteraciones mínimas necesarias según el Teorema 2.1: 14
La aproximación de la raíz es: 1.3247184753417969
```