Tarea 04: Ejercicios Unidad 02-A | Bissección

Métodos Numéricos

Christopher Criollo

2025-11-06

Tabla de Contenidos

1 CONJUNTO DE EJERCICIOS **2**

1.1 **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. 2

1.2 **2. a) Dibuje las gráficas para $y = x$ y $y = \sin x$.** 3

1.3 **3. a) Dibuje las gráficas para $y = \tan x$** 5

1.4 **4. a) Dibuje las gráficas para $y = x^2 - 1$ y $y = e^{1-x^2}$** 6

1.5 **5. Sea $f(x) = (x+3)(x+1)^2x(x-1)^3(x-3)$. ¿En qué cero de f converge el método de bisección cuando se aplica en los siguientes intervalos?** 9

2 Ejercicios aplicados **11**

2.1 **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 *h* a partir de la parte superior, el volumen □ de agua es** 11

2.2 **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 □ cae desde una altura □ y que la altura del objeto después de □ segundos es** . . 12

3 Ejercicios teóricos **12**

3.1 **1. Use el teorema 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.** 12

3.2 **2. La función definida por $f(x) = \sin \pi x$ tiene ceros en cada entero. Muestre cuando −1 < □ < 0 y 2 < □ < 3, el método de bisección converge a:** 13

1 CONJUNTO DE EJERCICIOS

1.1 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]

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

def sign(x: float) -> int:
    if x > 0:
        return 1
    elif x < 0:
        return -1
    else:
        return 0

from typing import Callable

def bisection(
    a: float, b: float, *, equation: Callable[[float], float], tol: float, N: int
) -> tuple[float, int] | None:
    i = 1

    assert a < b, "'a' no es menor a 'b', el inervalo no es valido."

    assert (
        equation(a) * equation(b) < 0
    ), "La función no cambia de signo en el intervalo."

    Fa = equation(a)
    p = a
    for i in range(N):
        p = a + (b - a) / 2
        FP = equation(p)

        if FP == 0 or (b - a) / 2 < tol:
            return p, i

        if sign(Fa) * sign(FP) > 0:
            a = p
            Fa = FP
```

```
        else:
            b = p

    return p, i

r1, i = bisection(a = 0, b = 1, equation = lambda x : x**3 - 7*x**2 + 14*x - 6, tol = 10
print(f'El resultado de la biseccion da la raíz {r1} en {i} iteraciones')
```

```
El resultado de la biseccion da la raíz 0.5859375 en 6 iteraciones
```

b) [1 , 3.2]

```
r2, i = bisection(a = 1, b = 3.2, equation = lambda x : x**3 - 7*x**2 + 14*x - 6, tol =
print(f'El resultado de la biseccion da la raíz {r2} en {i} iteraciones')
```

```
El resultado de la biseccion da la raíz 3.0023437500000005 en 7 iteraciones
```

c) [3.2 , 4]

```
r3, i = bisection(a = 3.2, b = 4, equation = lambda x : x**3 - 7*x**2 + 14*x - 6, tol =
print(f'El resultado de la biseccion da la raíz {r3} en {i} iteraciones')
```

```
El resultado de la biseccion da la raíz 3.41875 en 6 iteraciones
```

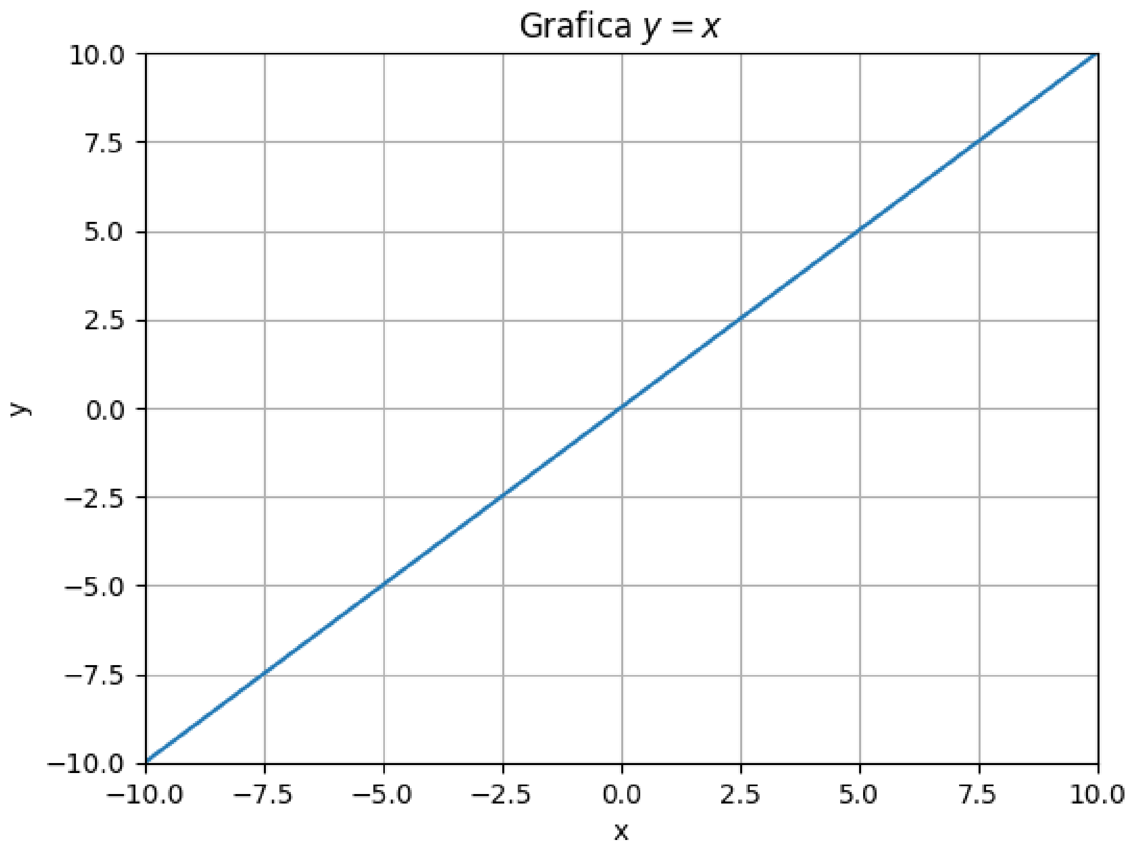
1.2 2. a) Dibuje las gráficas para $y = x$ y $y = \sin x$.

Gráfica de la función $y = x$.

```
X = np.linspace(-10, 10, 100)

plt.plot(X, X) # x = y

plt.xlabel('x')
plt.ylabel('y')
plt.title('Grafica $y = x$')
ax = plt.gca()
ax.set_ylim([-10, 10])
ax.set_xlim([-10, 10])
plt.grid(True)
plt.show()
```

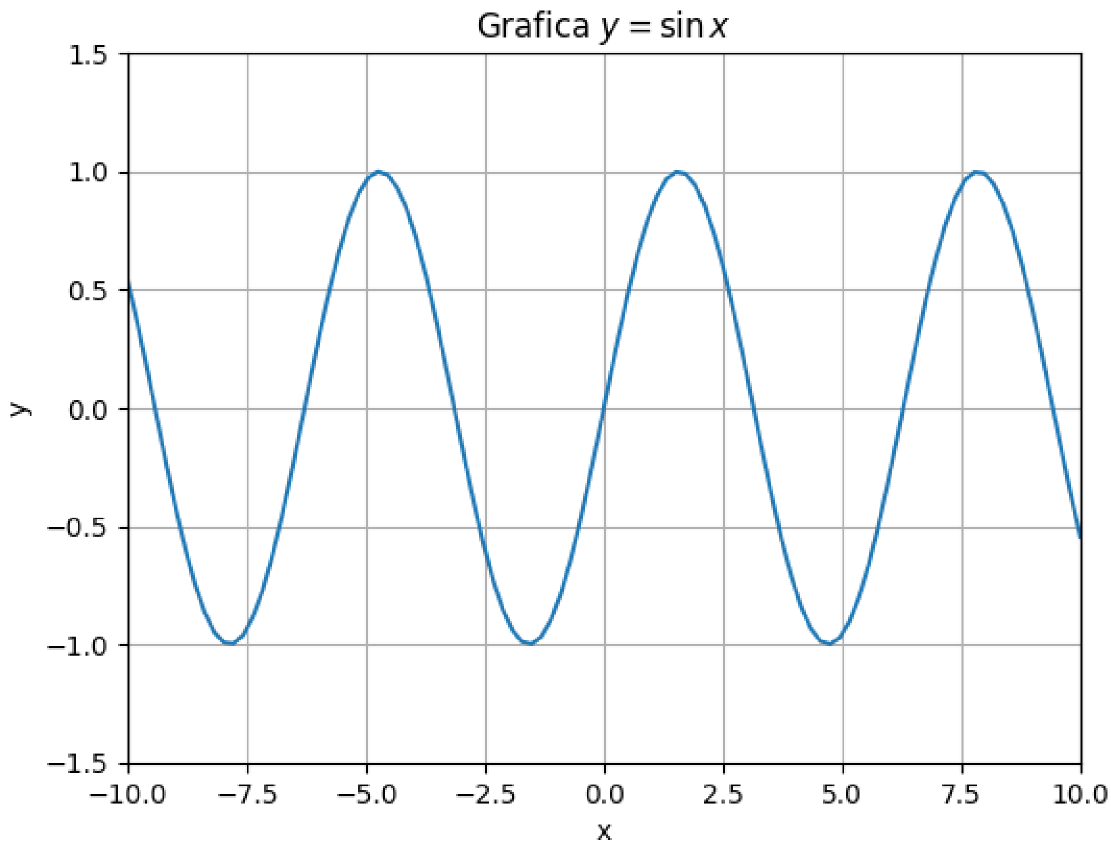


Gráfica de la función $y = \sin x$.

```
Y = [(lambda x : math.sin(x))(x) for x in X]

plt.plot(X, Y)

plt.xlabel('x')
plt.ylabel('y')
plt.title('Grafica $y = \sin{x}$')
ax = plt.gca()
ax.set_ylim([-1.5, 1.5])
ax.set_xlim([-10, 10])
plt.grid(True)
plt.show()
```



b) Use el método de bisección para encontrar soluciones precisas dentro de 10^{-5} para el primer valor positivo de x con $y = 2\sin x$.

```
r, i = bisection(a = 0.2, b = 3.5,
                 equation = lambda x : 2*math.sin(x), tol = 10**(-5), N = 20)
print(f'La biseccion da resultado la raíz {r} en {i} iteraciones')
```

```
La biseccion da resultado la raíz 3.141586494445801 en 18 iteraciones
```

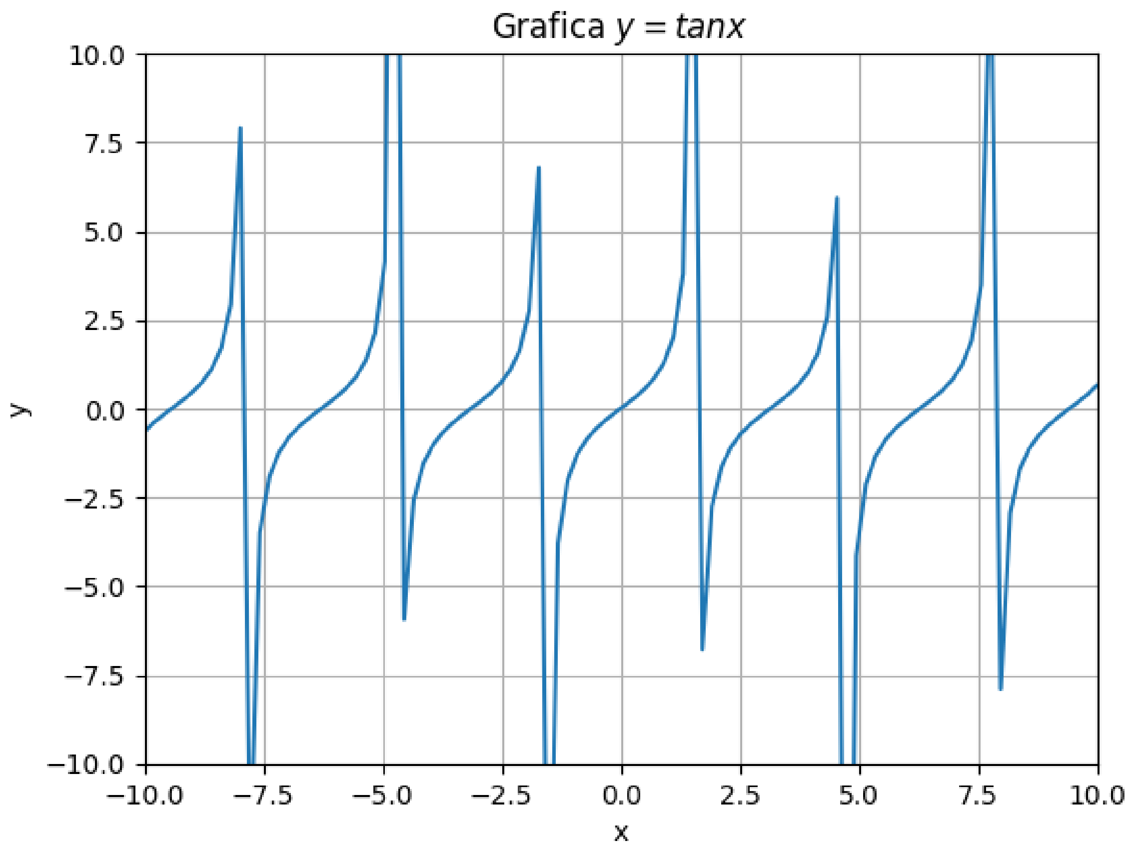
1.3 3. a) Dibuje las gráficas para $y = \tan x$

```
Y = np.tan(X)

plt.plot(X, Y)

plt.xlabel('x')
plt.ylabel('y')
plt.title('Grafica $y = tan{x}$')
```

```
ax = plt.gca()
ax.set_ylim([-10, 10])
ax.set_xlim([-10, 10])
plt.grid(True)
plt.show()
```



b) Use el método de bisección para encontrar una aproximación dentro de 10 para el primer valor positivo de x con $y = tan(x)$.

```
r, i = bisection(a = -1.2, b = 1,
                 equation = lambda x : math.tan(x), tol = 10**(-5), N = 20)
print(f'La biseccion da resultado la raíz {r} en {i} iteraciones')
```

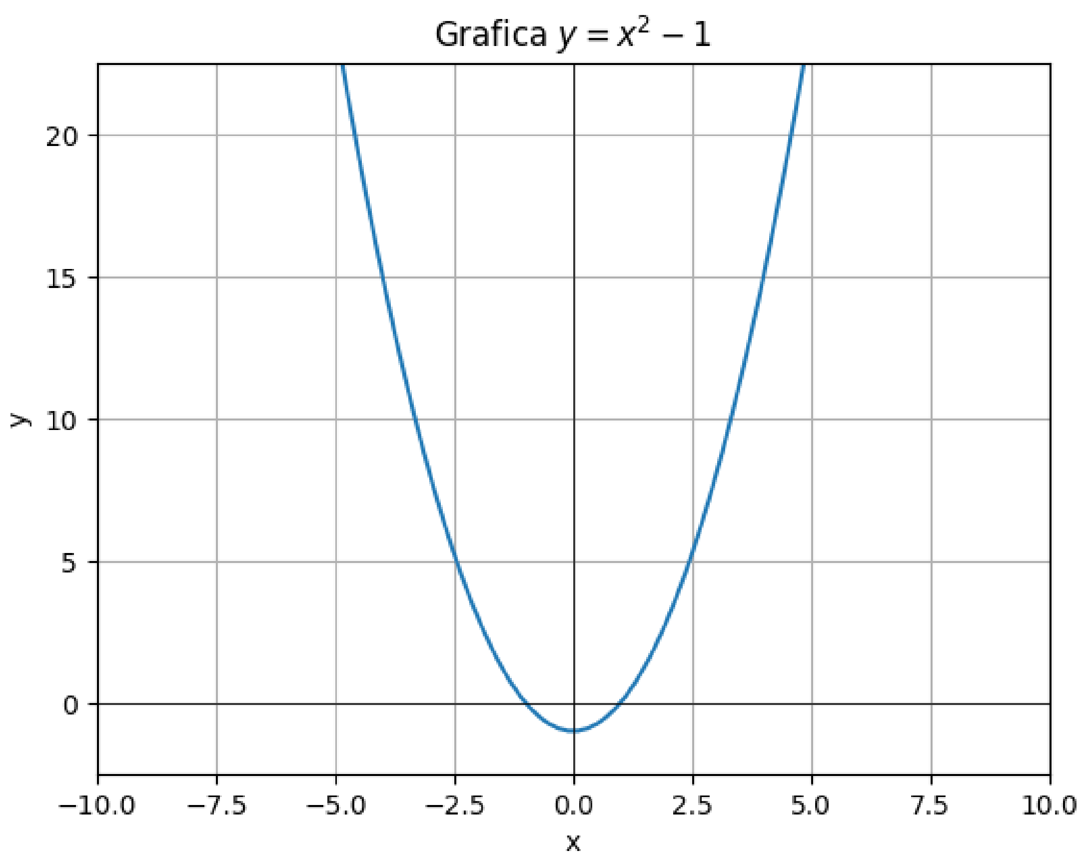
```
La biseccion da resultado la raíz -5.340576171753884e-06 en 17 iteraciones
```

1.4 4. a) Dibuje las gráficas para $y = x^2 - 1$ y $y = e^{1-x^2}$

Grafica de la función $y = x^2 - 1$:

```
X = np.linspace(-10, 10, 100)
Y = [(lambda x : x**2 - 1)(x) for x in X]

plt.plot(X, Y)
plt.xlabel('x')
plt.ylabel('y')
plt.title('Grafica $y = x^2 - 1$')
ax = plt.gca()
ax.set_ylim([-2.5, 22.5])
ax.set_xlim([-10, 10])
ax.axhline(0, color='black',linewidth=0.5)
ax.axvline(0, color='black',linewidth=0.5)
plt.grid(True)
plt.show()
```

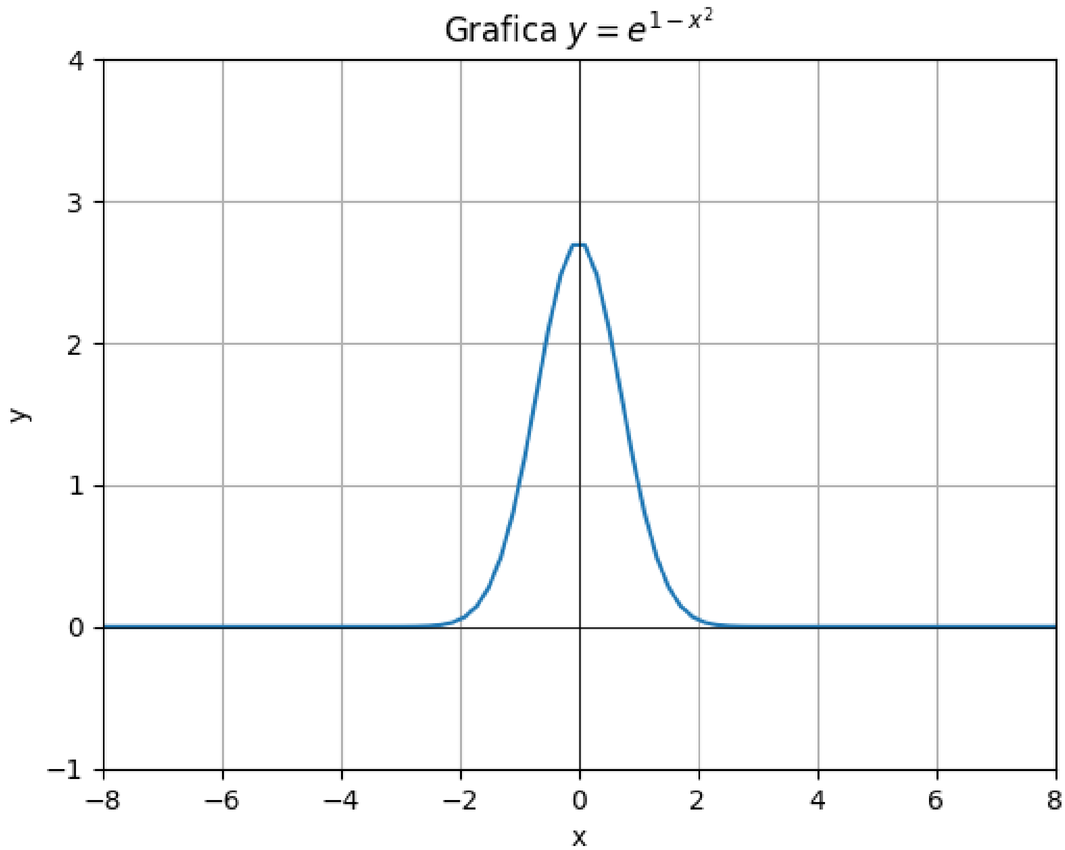


Gráfica para la función $y = e^{1-x^2}$:

```
Y = [(lambda x : math.exp(1 - x**2))(x) for x in X]

plt.plot(X, Y)
```

```
plt.xlabel('x')
plt.ylabel('y')
plt.title('Grafica $y = e^{1 - x^2}$')
ax = plt.gca()
ax.set_ylim([-1, 4])
ax.set_xlim([-8, 8])
ax.axhline(0, color='black',linewidth=0.5)
ax.axvline(0, color='black',linewidth=0.5)
plt.grid(True)
plt.show()
```



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

```
r, i = bisection(a = -2, b = 0, equation = lambda x : x**2 - math.exp(1 - x**2) - 1, tol
print(f'El resultado de biseccion da la raíz {r} en {i} iteraciones')
```

```
El resultado de biseccion da la raíz -1.2509765625 en 10 iteraciones
```

1.5 5. Sea $f(x) = (x+3)(x+1)^2 x(x-1)^3(x-3)$. ¿En qué cero de f converge el método de bisección cuando se aplica en los siguientes intervalos?

a) (-1.5 ; 2.5)

```
r, i = bisection(a = -1.5, b = 2.5,
                 equation = lambda x : (x + 3) * (x + 1)**2 * x * (x - 1)**3 * (x - 3),
                 tol = 10**(-3), N = 20)
print(f'La biseccion dio como resultado la raíz {r} en {i} iteraciones')
```

```
AssertionError: La función no cambia de signo en el intervalo.
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
Cell In[18], line 1
----> 1 r, i = bisection(a = -1.5, b = 2.5,
      2                  equation = lambda x : (x + 3) * (x + 1)**2 * x * (x - 1)**3 * (
      3                  tol = 10**(-3), N = 20)
      4 print(f'La biseccion dio como resultado la raíz {r} en {i} iteraciones')
Cell In[1], line 23, in bisection(a, b, equation, tol, N)
     18 i = 1
     20 assert a < b, "'a' no es menor a 'b', el inervalo no es valido."
     22 assert (
---> 23     equation(a) * equation(b) < 0
     24 ), "La función no cambia de signo en el intervalo."
     26 Fa = equation(a)
     27 p = a
AssertionError: La función no cambia de signo en el intervalo.
```

Dado que $f(-1.5)$ y $f(2.5)$ tienen el mismo signo (ambos son negativos), no se cumple la condición $f(a) \cdot f(b) < 0$.

b. (-0.5 ; 2.4):

```
r, i = bisection(a = -0.5, b = 2.4,
                 equation = lambda x : (x + 3) * (x + 1)**2 * x * (x - 1)**3 * (x - 3),
                 tol = 10**(-3), N = 20)
print(f'La biseccion dio como resultado la raíz {r} en {i} iteraciones')
```

```
AssertionError: La función no cambia de signo en el intervalo.
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
Cell In[19], line 1
----> 1 r, i = bisection(a = -0.5, b = 2.4,
      2                  equation = lambda x : (x + 3) * (x + 1)**2 * x * (x - 1)**3 * (
```

```
      3                   tol = 10**(-3), N = 20)
      4 print(f'La biseccion dio como resultado la raíz {r} en {i} iteraciones')
Cell In[1], line 23, in bisection(a, b, equation, tol, N)
     18 i = 1
     20 assert a < b, "'a' no es menor a 'b', el inervalo no es valido."
     22 assert (
---> 23     equation(a) * equation(b) < 0
     24 ), "La función no cambia de signo en el intervalo."
     26 Fa = equation(a)
     27 p = a
AssertionError: La función no cambia de signo en el intervalo.
```

c. (-0.5 ; 3):

```
r, i = bisection(a = -0.5, b = 3,
                 equation = lambda x : (x + 3) * (x + 1)**2 * x * (x - 1)**3 * (x - 3),
                 tol = 10**(-3), N = 20)
print(f'La biseccion dio como resultado la raíz {r} en {i} iteraciones')
```

```
AssertionError: La función no cambia de signo en el intervalo.
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
Cell In[20], line 1
----> 1 r, i = bisection(a = -0.5, b = 3,
      2                   equation = lambda x : (x + 3) * (x + 1)**2 * x * (x - 1)**3 * (
      3                   tol = 10**(-3), N = 20)
      4 print(f'La biseccion dio como resultado la raíz {r} en {i} iteraciones')
Cell In[1], line 23, in bisection(a, b, equation, tol, N)
     18 i = 1
     20 assert a < b, "'a' no es menor a 'b', el inervalo no es valido."
     22 assert (
---> 23     equation(a) * equation(b) < 0
     24 ), "La función no cambia de signo en el intervalo."
     26 Fa = equation(a)
     27 p = a
AssertionError: La función no cambia de signo en el intervalo.
```

d. (-3 ; -0.5):

```
r, i = bisection(a = -3, b = -0.5,
                 equation = lambda x : (x + 3) * (x + 1)**2 * x * (x - 1)**3 * (x - 3),
                 tol = 10**(-3), N = 20)
print(f'La biseccion dio como resultado la raíz {r} en {i} iteraciones')
```

2 Ejercicios aplicados

2.1 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 de agua es

$$V = L[0.5\pi r^2 - r^2 \arcsin \frac{h}{r} - h(r^2 - h^2)^{\frac{1}{2}}]$$

Suponga que □ = 10 □□, □ = 1 □□ y □ =12.4 □□. Encuentre la profundidad del agua en el abrevadero dentro de 0.01 □□.

En este problema, 0.01 cm representa el error absoluto admisible en el cálculo de la altura h.

Paso 1: Sustitución en la Ecuación Original

$$12.4 = 10[0.5\pi(1)^2 - 1^2 \arcsin \frac{h}{1} - h(1^2 - h^2)^{\frac{1}{2}}]$$

Paso 2: Reformulación para Aplicar Método de Bisección Para encontrar las raíces escribimos la ecuación igualándola a cero:

$$0 = 10[0.5\pi(1)^2 - 1^2 \arcsin \frac{h}{1} - h(1^2 - h^2)^{\frac{1}{2}}] - 12.4$$

Esta forma es necesaria para aplicar el método de bisección, ya que buscamos el valor de h que satisface la ecuación.

```
r, i = bisection(a = 0, b = 1,
                 equation = lambda x : 10 * (0.5 * math.pi - math.asin(x) - x * (1 - x**
                 tol = 0.01, N = 20)
print(f'La profundidad del agua (h) es aproximadamente: {r} cm, calculado en {i} iteraci
```

```
La profundidad del agua (h) es aproximadamente: 0.1640625 cm, calculado en 6 iteraciones
```

2.2 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 cae desde una altura y que la altura del objeto después de segundos es

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

donde □ =9.81 $\frac{m}{s^2}$ y □ representa el coeficiente de la resistencia del aire en $\frac{Ns}{m}$. Suponga $s_0 = 300m$, $m = 25kg$ y $k = 0.1\frac{Ns}{m}$. Encuentre, dentro de 0.01 □□□□□□□□, el tiempo que tarda un cuarto de kg en golpear el piso.

Si 0.01 segundos es el error absoluto admisible para la raíz. Se remplaza los datos dados en el enunciado para la fórmula, tal que la ecuación esté respecto al tiempo.

$0 = 300 - \frac{(25)(9.81)}{0.1}t + \frac{(25)^2(9.81)}{(0.1)^2}(1 - e^{\frac{-(0.1)t}{25}})$

Una sustitución adicional es $s(t) = 0$ debido a que esta función representa la altura en la que el objeto se encuentra en cierto momento, pero el problema nos pide el tiempo para cuando colisione con el suelo. Tomando como el sistema de referencia para el eje y el suelo, entonces la altura es cero.

Biseccion:

```
r, i = bisection(a = 7, b = 8,
                 equation = lambda x : 300 - (25*9.81*x)/0.1 + (625*9.81)*(1 - math.exp(
                 tol = 0.01, N = 20)
print(f'El tiempo en tocar el suelo (t) es aproximadamente: {r} segundos, calculado en {
```

```
El tiempo en tocar el suelo (t) es aproximadamente: 7.8671875 segundos, calculado en 6 i
```

3 Ejercicios teóricos

3.1 1. Use el teorema 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.

Método de Biseccion con los parametros:

```python
r, i = bisection(a = 1, b = 2,
                 equation = lambda x : x**3 - x - 1,
                 tol = 10**(-4), N = 20)
print(f'La biseccion da resultado la raíz {r} en {i} iteraciones')
```

```
La biseccion da resultado la raíz 1.32476806640625 en 13 iteraciones
```

3.2 2. La función definida por $f(x) = \sin \pi x$ tiene ceros en cada entero. Muestre cuando $-1 < \quad < 0$ y $2 < \quad < 3$, el método de bisección converge a:

a. 0,□□ □ + □ < 2

Para que se cumpla □ + □ < 2, tomamos como valores: a = -0.99 y b = 2.1 Comprobamos cual es la raíz a la que converge en este caso:

```python
r, i = bisection(a = -0.99, b = 2.1,
                 equation = lambda x : math.sin(math.pi * x),
                 tol = 10**(-3), N = 20)
print(f'La biseccion da resultado la raíz {r} en {i} iteraciones')
```

```
La biseccion da resultado la raíz 0.0005200195312499633 en 11 iteraciones
```

b. 2,□□ □ + □ > 2

Para que se cumpla □ + □ > 2, tomamos como valores: a = -0.1 y b = 2.9 Comprobamos cual es la raíz a la que converge en este caso:

```python
r, i = bisection(a = -0.1, b = 2.9,
                 equation = lambda x : math.sin(math.pi * x),
                 tol = 10**(-3), N = 20)
print(f'La biseccion da resultado la raíz {r} en {i} iteraciones')
```

```
La biseccion da resultado la raíz 1.999853515625 en 11 iteraciones
```

c. 1, □□ □ +□ = 2

Para que se cumpla □ + □ = 2, tomamos como valores: a = -0.5 y b = 2.5 Comprobamos cual es la raíz a la que converge en este caso:

```python
r, i = bisection(a = -0.5, b = 2.5,
                 equation = lambda x : math.sin(math.pi * x),
                 tol = 10**(-3), N = 20)
print(f'La biseccion da resultado la raíz {r} en {i} iteraciones')
```

```
La biseccion dio como resultado la raíz 0.000244140625 en 11 iteraciones
```