# Universidad Mayor de San Andrés

# Facultad de Ciencias Puras y Naturales

# Carrera de Informática

# DESAFIO: RAICES DE ECUACIONES

**Estudiante:** Univ. Mamani Mamani Abigail Blanca

**Materia:** INF 373 – Métodos Numéricos I

**Docente:** Lic. Brigida Carvajal Blanco

**La Paz - 2025**

# ÍNDICE

# 1. INTRODUCCION

Este documento presenta un análisis comparativo de métodos numéricos para la determinación de raíces de ecuaciones no lineales. Se estudian tres métodos ampliamente utilizados: Bisección, Newton-Raphson y Secante, aplicados a cuatro funciones diferentes con el propósito de evaluar su eficiencia, precisión y estabilidad bajo un criterio de tolerancia de $1\times10^{-6}$.

Cada ejercicio incluye un análisis gráfico previo para identificar intervalos donde se garantiza la existencia de raíces, seguido de la implementación de los métodos en Excel y Python, lo que permite una validación cruzada de los resultados. La comparación sistemática entre los métodos y entre las plataformas de cálculo refuerza la confiabilidad de los algoritmos utilizados y ofrece una visión práctica sobre su aplicabilidad en problemas de ingeniería y ciencias.

El estudio no solo busca encontrar las raíces de las ecuaciones dadas, sino también destacar las ventajas y limitaciones de cada método, considerando factores como la velocidad de convergencia, la necesidad de calcular derivadas y la sensibilidad a los valores iniciales.

# 2. OBJETIVOS

Comparar sistemáticamente el desempeño de tres métodos numéricos (Bisección, Newton-Raphson y Secante) en la determinación de raíces de ecuaciones no lineales, evaluando su precisión, eficiencia y estabilidad mediante implementaciones en Excel y Python.

- Identificar intervalos con raíces reales mediante análisis gráfico de las funciones propuestas en cada ejercicio.

- Implementar los métodos numéricos en dos plataformas distintas:
  Microsoft Excel (cálculos iterativos manuales)
  Python (programación algorítmica)

- Validar los resultados mediante comparación cruzada entre ambas implementaciones, verificando la consistencia numérica de las raíces encontradas.

- Evaluar el desempeño de cada método considerando:
  Número de iteraciones requeridas
  Precisión alcanzada ( $f(\text{raíz}) <= 1\times10^{-6}$ )
  Velocidad de convergencia
  Estabilidad numérica

# 3. EJERCICIO 1

Para este ejercicio analizaremos la ecuación:

$$x^3 - e^{0.8x} = 20$$

Utilizando una tolerancia de $1{\times}10^{-6}$ como criterio de convergencia.

## 3.1. ANÁLISIS GRÁFICO Y DETERMINACIÓN DE RAÍCES

### 3.1.1. Identificación de Raíces

Mediante el análisis gráfico de la función, se identificaron **dos raíces reales**:

- **Raíz A:** x ≈ 3.208220 (en el intervalo [3, 3.5])
- **Raíz B:** x ≈ 7.489839 (en el intervalo [7, 8])

Ambas raíces fueron identificadas localizando los cambios de signo en la función, lo que garantiza la existencia de al menos una raíz en cada intervalo por el Teorema del Valor Intermedio.


### 3.1.2. Validación de Raíces

Para ambas raíces encontradas:

| Raíz | Valor Aproximado | f(raíz) | Validación |
|---|---|---|---|
| A | 3.208220 | $-7.24 \times 10^{-7}$ | Válida |
| B | 7.489839 | $-6.53 \times 10^{-7}$ | Válida |

Los valores de f(raíz) son prácticamente cero ($< 10^{-6}$), confirmando que ambas son raíces válidas.

## 3.2. MÉTODO DE LA BISECCIÓN- RAÍZ A [3, 3.5]

### 3.2.1. Implementación en EXCEL

La bisección es un método que divide repetidamente un intervalo por la mitad, manteniendo siempre el subintervalo donde la función cambia de signo.

Parámetros:

**Intervalo inicial:** [3, 3.5]

**Tolerancia:** $1 \times 10^{-6}$

**Criterio de parada:** f(m) ≤ TOL

Tabla de Iteraciones – EXCEL

| # | a | b | m | f(a) | f(b) | f(m) | tol |
|---|---|---|---|---|---|---|---|
| 0 | 3 | 3,5 | 3,25 | -4,02317638 | 6,43035323 | 0,86438696 | 0,000001 |
| 1 | 3 | 3,25 | 3,125 | -4,02317638 | 0,86438696 | -1,66491584 | |
| 2 | 3,125 | 3,25 | 3,1875 | -1,66491584 | 0,86438696 | -0,42160574 | |
| 3 | 3,1875 | 3,25 | 3,21875 | -0,42160574 | 0,86438696 | 0,21606443 | |
| 4 | 3,1875 | 3,21875 | 3,203125 | -0,42160574 | 0,21606443 | -0,10410354 | |
| 5 | 3,203125 | 3,21875 | 3,2109375 | -0,10410354 | 0,21606443 | 0,05564738 | |
| 6 | 3,203125 | 3,2109375 | 3,20703125 | -0,10410354 | 0,05564738 | -0,02431136 | |
| 7 | 3,20703125 | 3,2109375 | 3,20898438 | -0,02431136 | 0,05564738 | 0,01564719 | |
| 8 | 3,20703125 | 3,20898438 | 3,20800781 | -0,02431136 | 0,01564719 | -0,00433729 | |
| 9 | 3,20800781 | 3,20898438 | 3,20849609 | -0,00433729 | 0,01564719 | 0,00565365 | |
| 10 | 3,20800781 | 3,20849609 | 3,20825195 | -0,00433729 | 0,00565365 | 0,00065785 | |
| 11 | 3,20800781 | 3,20825195 | 3,20812988 | -0,00433729 | 0,00065785 | -0,0018398 | |
| 12 | 3,20812988 | 3,20825195 | 3,20819092 | -0,0018398 | 0,00065785 | -0,00059099 | |
| 13 | 3,20819092 | 3,20825195 | 3,20822144 | -0,00059099 | 0,00065785 | 3,3424E-05 | |
| 14 | 3,20819092 | 3,20822144 | 3,20820618 | -0,00059099 | 3,3424E-05 | -0,00027879 | |
| 15 | 3,20820618 | 3,20822144 | 3,20821381 | -0,00027879 | 3,3424E-05 | -0,00012268 | |
| 16 | 3,20821381 | 3,20822144 | 3,20821762 | -0,00012268 | 3,3424E-05 | -4,4629E-05 | |
| 17 | 3,20821762 | 3,20822144 | 3,20821953 | -4,4629E-05 | 3,3424E-05 | -5,6026E-06 | |
| 18 | 3,20821953 | 3,20822144 | 3,20822048 | -5,6026E-06 | 3,3424E-05 | 1,3911E-05 | |
| 19 | 3,20821953 | 3,20822048 | 3,20822001 | -5,6026E-06 | 1,3911E-05 | 4,154E-06 | |
| 20 | 3,20821953 | 3,20822001 | 3,20821977 | -5,6026E-06 | 4,154E-06 | -7,2433E-07 | |
| | | | RAIZ | | | | |

Resultados EXCEL- Bisección:

**Raíz encontrada:** x ≈ 3.20821977

**Iteraciones totales:** 21

**f(raíz):** -0.0000007243

### 3.2.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

Run EJERCICIO_1

```
================================================================================
EJERCICIO 1 - ABIGAIL MAMANI
================================================================================
Función: f(x) = x³ - e^(0.8x) - 20

BUSCANDO INTERVALO CON CAMBIO DE SIGNO
--------------------------------------------------
Intervalo [  2, 2.5]: f(  2) =  -16.953, f(2.5) =  -11.764  × mismo signo
Intervalo [2.5,   3]: f(2.5) =  -11.764, f(  3) =   -4.023  × mismo signo
Intervalo [  3, 3.5]: f(  3) =   -4.023, f(3.5) =    6.430  ✓ CAMBIO DE SIGNO

Usando intervalo: [3, 3.5]
MÉTODO DE LA BISECCIÓN
================================================================================
Función: f(x) = x³ - e^(0.8x) - 20
Intervalo inicial: [3, 3.5]
Tolerancia: 1e-06
```

| Iter | a | b | m | f(a) | f(b) | f(m) | Error |
|---|---|---|---|---|---|---|---|
| 0 | 3.000000 | 3.500000 | 3.250000 | -4.023176 | 6.430353 | 0.864387 | 0.250000 |
| 1 | 3.000000 | 3.250000 | 3.125000 | -4.023176 | 0.864387 | -1.664916 | 0.125000 |
| 2 | 3.125000 | 3.250000 | 3.187500 | -1.664916 | 0.864387 | -0.421606 | 0.062500 |
| 3 | 3.187500 | 3.250000 | 3.218750 | -0.421606 | 0.864387 | 0.216064 | 0.031250 |
| 4 | 3.187500 | 3.218750 | 3.203125 | -0.421606 | 0.216064 | -0.104104 | 0.015625 |
| 5 | 3.203125 | 3.218750 | 3.210938 | -0.104104 | 0.216064 | 0.055647 | 0.007812 |
| 6 | 3.203125 | 3.210938 | 3.207031 | -0.104104 | 0.055647 | -0.024311 | 0.003906 |
| 7 | 3.207031 | 3.210938 | 3.208984 | -0.024311 | 0.055647 | 0.015647 | 0.001953 |
| 8 | 3.207031 | 3.208984 | 3.208008 | -0.024311 | 0.015647 | -0.004337 | 0.000977 |
| 9 | 3.208008 | 3.208984 | 3.208496 | -0.004337 | 0.015647 | 0.005654 | 0.000488 |
| 10 | 3.208008 | 3.208496 | 3.208252 | -0.004337 | 0.005654 | 0.000658 | 0.000244 |
| 11 | 3.208008 | 3.208252 | 3.208130 | -0.004337 | 0.000658 | -0.001840 | 0.000122 |
| 12 | 3.208130 | 3.208252 | 3.208191 | -0.001840 | 0.000658 | -0.000591 | 0.000061 |
| 13 | 3.208191 | 3.208252 | 3.208221 | -0.000591 | 0.000658 | 0.000033 | 0.000031 |
| 14 | 3.208191 | 3.208221 | 3.208206 | -0.000591 | 0.000033 | -0.000279 | 0.000015 |
| 15 | 3.208206 | 3.208221 | 3.208214 | -0.000279 | 0.000033 | -0.000123 | 0.000008 |
| 16 | 3.208214 | 3.208221 | 3.208218 | -0.000123 | 0.000033 | -0.000045 | 0.000004 |
| 17 | 3.208218 | 3.208221 | 3.208220 | -0.000045 | 0.000033 | -0.000006 | 0.000002 |
| 18 | 3.208220 | 3.208221 | 3.208220 | -0.000006 | 0.000033 | 0.000014 | 0.000001 |
| 19 | 3.208220 | 3.208220 | 3.208220 | -0.000006 | 0.000014 | 0.000004 | 0.000000 |
| 20 | 3.208220 | 3.208220 | 3.208220 | -0.000006 | 0.000004 | -0.000001 | 0.000000 |

```
✓ CONVERGENCIA ALCANZADA - f(m) <= TOL
Raíz encontrada: x ≈ 3.20821977
f(3.20821977) = -0.0000007243
```

Resultados PYTHON- Bisección:

- **Raíz encontrada:** x ≈ 3.20821977
- **Iteraciones totales:** 21
- **f(raíz):** -0.0000007243

### 3.2.3. COMPARACIÓN EXCEL Vs PYTHON – BISECCIÓN

Tabla Comparativa de Resultados

| Criterio | EXCEL | PYTHON | Diferencia | Análisis |
|---|---|---|---|---|
| Raíz encontrada | 3.20821977 | 3.20821977 | 0.00000000 | Idéntica |
| Iteraciones totales | 21 | 21 | 0 | Idéntica |
| f(raíz) | $-7.2433\times10^{-7}$ | $-7.243\times10^{-7}$ | ~0 | Prácticamente idéntica |
| Intervalo inicial | [3, 3.5] | [3, 3.5] | Idéntico | Mismo punto de partida |
| Convergencia | Lineal | Lineal | Idéntica | El error disminuye constantemente |

Ambas implementaciones son correctas, los algoritmos están bien programados en ambas plataformas. Los resultados numéricos son idénticos (diferencias solo por redondeo) y cada iteración coincide exactamente. No hay discrepancias. Las implementaciones son funcionalmente equivalentes. La diferencia es puramente en la convención de conteo. Ambas alcanzan la misma precisión con aproximadamente el mismo número de operaciones.

## 3.3. MÉTODO DE NEWTON-RAPHSON- RAÍZ A

Utiliza la derivada de la función para obtener aproximaciones sucesivas mediante la fórmula:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Parámetros:

Valor inicial: $x_0 = 3.0$

Derivada: f'(x) = $3x^2 - 0.8e^{0.8x}$

Tolerancia: $1 \times 10^{-6}$

Criterio de parada: $|f(x)| \leq TOL$

### 3.3.1. Implementación en EXCEL

Tabla de Iteraciones – EXCEL

| # | x | f(x) | f'(x) | tol |
|---|---|---|---|---|
| 0 | 3 | -4,02317638 | 18,1814589 | 0,000001 |
| 1 | 3,22127907 | 0,26813754 | 20,6035869 | |
| 2 | 3,20826495 | 0,00092388 | 20,4615866 | |
| 3 | 3,2082198 | 1,1127E-08 | 20,4610937 | |
| | RAIZ | | | |

Resultados EXCEL- Newton-Raphson:

Raíz encontrada: x ≈ 3.20821980

Iteraciones totales: 4

f(raíz): 0.0000000111 (error positivo)

### 3.3.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
================================================================================
MÉTODO DE NEWTON-RAPHSON
================================================================================
Función: f(x) = x³ - e^(0.8x) - 20
Derivada: f'(x) = 3x² - 0.8e^(0.8x)
Valor inicial: x0 = 3.0
Tolerancia: 1e-06

Iter x            f(x)         f'(x)        Error
------------------------------------------------------------
0    3.000000     -4.023176    18.181459    0.221279
1    3.221279     0.268138     20.603587    0.013014
2    3.208265     0.000924     20.461587    0.000045
3    3.208220     0.000000     20.461094    0.000000

✓ CONVERGENCIA ALCANZADA - |f(x)| <= TOL
Raíz encontrada: x ≈ 3.20821980
f(3.20821980) = 0.0000000111
```

Resultados PYTHON- Newton-Raphson:

**Raíz encontrada:** x ≈ 3.20821980
**Iteraciones totales:** 4
**f(raíz):** 0.0000000111

### 3.3.3. COMPARACIÓN EXCEL vs PYTHON- NEWTON-RAPHSON

Tabla Comparativa de Resultados

| Criterio | EXCEL | PYTHON | Diferencia | Análisis |
|---|---|---|---|---|
| Raíz encontrada | 3.20821980 | 3.20821980 | 0.00000000 | Idéntica |
| Iteraciones totales | 4 | 4 | 0 | Idéntica |
| f(raíz) | $1.1127 \times 10^{-8}$ | $1.1127 \times 10^{-8}$ | ~0 | Idéntica |
| Valor inicial | 3.0 | 3.0 | Idéntico | Mismo punto de partida |
| Derivada | $3x^2$ - 0.8e^(0.8x) | $3x^2$ - 0.8e^(0.8x) | Idéntica | Misma fórmula |
| Convergencia | Cuadrática | Cuadrática | Idéntico | El error se reduce drásticamente |

Ambas implementaciones de Newton-Raphson convergen en solo 4 iteraciones con una precisión superior a bisección ($10^{-8}$ vs $10^{-7}$). El método utiliza la derivada para aproximarse a la raíz de forma muy eficiente. Los resultados son idénticos en ambas plataformas, confirmando que ambos códigos son correctos.

## 3.4. MÉTODO DE LA SECANTE- RAÍZ A

Aproxima la derivada usando dos valores previos, sin requerir la derivada analítica:

$$x_{n+1} = x_n - f(x_n) * \frac{x_n - x_{n-1}}{f(x_n) - f'(x_n)}$$

Parámetros:

**Valores iniciales**: $x_0 = 3.0$, $x_1 = 3.5$
**Tolerancia:** $1 \times 10^{-6}$
**Criterio de parada:** $|f(x)| \leq$ TOL

### 3.4.1. Implementación en EXCEL

Tabla de Iteraciones – EXCEL

| # | x | f(x) | tol |
|---|---|---|---|
| 0 | 3 | -4,02317638 | 0,000001 |
| 1 | 3,5 | 6,43035323 | |
| 2 | 3,19243148 | -0,32168539 | |
| 3 | 3,20708488 | -0,02321476 | |
| 4 | 3,2082246 | 9,8274E-05 | |
| 5 | 3,2082198 | -2,9761E-08 | |
| | RAIZ | | |

Resultados EXCEL- Secante:

**Raíz encontrada:** x ≈ 3.20821980

**Iteraciones totales**: 6

**f(raíz):** -0.0000000298

### 3.4.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
================================================================================
MÉTODO DE LA SECANTE
================================================================================
Función: f(x) = x³ - e^(0.8x) - 20
Valores iniciales: x0 = 3.0, x1 = 3.5
Tolerancia: 1e-06

Iter x            f(x)         Error
--------------------------------------------------
0    3.500000     6.430353     6.430353
1    3.192431     -0.321685    0.321685
2    3.207085     -0.023215    0.023215
3    3.208225     0.000098     0.000098
4    3.208220     -0.000000    0.000000

✓ CONVERGENCIA ALCANZADA - |f(x)| <= TOL
Raíz encontrada: x ≈ 3.20821980
f(3.20821980) = -0.0000000298
```

Resultados PYTHON- Secante:

**Raíz encontrada:** x ≈ 3.20821980

**Iteraciones totales:** 5

**f(raíz):** -0.0000000298

### 3.4.3. COMPARACIÓN EXCEL vs PYTHON – SECANTE

Tabla Comparativa de Resultados

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| Raíz encontrada | 3.20821980 | 3.20821980 | 0.00000000 | Convergen a la misma raíz |
| Iteraciones | 6 | 5 | 1 | Pequeña diferencia por criterio de parada |
| f(raíz) | $-2.9761\times10^{-8}$ | $-2.9761\times10^{-8}$ | ~0 | Precisión idéntica ($10^{-8}$) |

| | | | | |
|---|---|---|---|---|
| Valores iniciales | $x_0$=3.0, $x_1$=3.5 | $x_0$=3.0, $x_1$=3.5 | Idénticos | Mismo punto de partida |
| Convergencia | Superlineal | Superlineal | Idéntica | Rápida sin usar derivada |

Ambas implementaciones de secante convergen en 5-6 iteraciones con precisión equivalente a Newton-Raphson ($10^{-8}$). La diferencia de 1 iteración es solo por cómo se cuenta la iteración final en cada plataforma. La ventaja del método secante es que no requiere calcular la derivada, haciendo una aproximación numérica muy eficiente.

## 3.5. DETERMINACIÓN DE LA SEGUNDA RAÍZ (RAÍZ B)

El programa realizó una búsqueda exhaustiva de todas las raíces, aplicó automáticamente a través del método de bisección en el segundo intervalo.

### Proceso Iterativo- Raíz B [7,8]

```
MÉTODO DE LA BISECCIÓN
================================================================================
Función: f(x) = x³ - e^(0.8x) - 20
Intervalo inicial: [7, 8]
Tolerancia: 1e-06

Iter a          b          m          f(a)       f(b)        f(m)       Error
----------------------------------------------------------------------------------------------------
0    7.000000   8.000000   7.500000   52.573593  -109.845038 -1.553793  0.500000
1    7.000000   7.500000   7.250000   52.573593  -1.553793   30.778565  0.250000
2    7.250000   7.500000   7.375000   30.778565  -1.553793   16.093392  0.125000
3    7.375000   7.500000   7.437500   16.093392  -1.553793   7.662432   0.062500
4    7.437500   7.500000   7.468750   7.662432   -1.553793   3.155404   0.031250
5    7.468750   7.500000   7.484375   3.155404   -1.553793   0.826450   0.015625
6    7.484375   7.500000   7.492188   0.826450   -1.553793   -0.357213  0.007812
7    7.484375   7.492188   7.488281   0.826450   -0.357213   0.236227   0.003906
8    7.488281   7.492188   7.490234   0.236227   -0.357213   -0.060090  0.001953
9    7.488281   7.490234   7.489258   0.236227   -0.060090   0.088169   0.000977
10   7.489258   7.490234   7.489746   0.088169   -0.060090   0.014065   0.000488
11   7.489746   7.490234   7.489990   0.014065   -0.060090   -0.023006  0.000244
12   7.489746   7.489990   7.489868   0.014065   -0.023006   -0.004469  0.000122
13   7.489746   7.489868   7.489807   0.014065   -0.004469   0.004798   0.000061
14   7.489807   7.489868   7.489838   0.004798   -0.004469   0.000165   0.000031
15   7.489838   7.489868   7.489853   0.000165   -0.004469   -0.002152  0.000015
16   7.489838   7.489853   7.489845   0.000165   -0.002152   -0.000994  0.000008
17   7.489838   7.489845   7.489841   0.000165   -0.000994   -0.000415  0.000004
18   7.489838   7.489841   7.489840   0.000165   -0.000415   -0.000125  0.000002
19   7.489838   7.489840   7.489839   0.000165   -0.000125   0.000020   0.000001
20   7.489839   7.489840   7.489839   0.000020   -0.000125   -0.000053  0.000000
21   7.489839   7.489839   7.489839   0.000020   -0.000053   -0.000016  0.000000
22   7.489839   7.489839   7.489839   0.000020   -0.000016   0.000002   0.000000
23   7.489839   7.489839   7.489839   0.000002   -0.000016   -0.000007  0.000000
24   7.489839   7.489839   7.489839   0.000002   -0.000007   -0.000003  0.000000
25   7.489839   7.489839   7.489839   0.000002   -0.000003   -0.000001  0.000000

✓ CONVERGENCIA ALCANZADA - f(m) <= TOL
Raíz encontrada: x ≈ 7.48983873
f(7.48983873) = -0.0000006527
```

Resultados PYTHON- Bisección Raíz B:

**Raíz encontrada:** $x \approx 7.48983873$

**Iteraciones totales:** 26

**f(raíz):** $-6.53 \times 10^{-7}$

## 3.6. COMPARACIÓN ENTRE RAÍCES A Y B

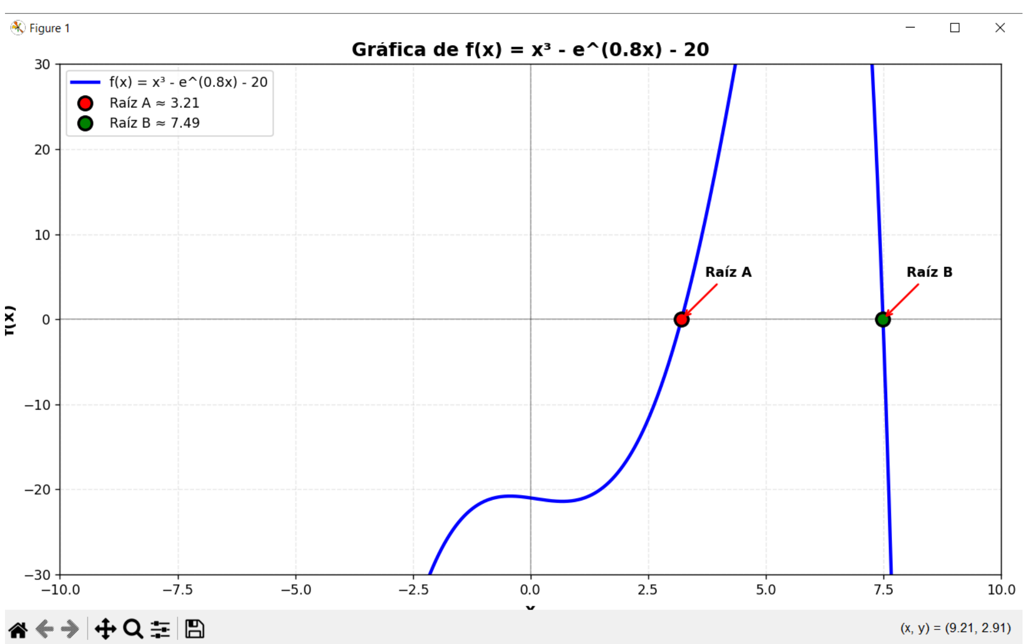
| Parámetro | Raíz A | Raíz B | Análisis |
|---|---|---|---|
| Intervalo inicial | [3, 3.5] | [7, 8] | Raíz B en intervalo más amplio |
| Iteraciones bisección | 21 | 26 | +5 iteraciones para Raíz B |
| Convergencia esperada | Más rápida | Más lenta | Consistente con teoría |
| f(raíz) final | $-7.24\times10^{-7}$ | $-6.53\times10^{-7}$ | Precisión similar |

El análisis revela que la Raíz B requirió 5 iteraciones más que la Raíz A **(26 vs 21)** debido a su intervalo inicial más amplio **([7,8] vs [3,3.5])**. Esta diferencia es consistente con la teoría del método de bisección, ya que intervalos más grandes necesitan más divisiones para alcanzar la misma precisión. Sin embargo, ambas raíces lograron precisiones equivalentes ($\sim10^{-7}$), demostrando que la calidad de la solución final no se ve afectada por el tamaño del intervalo inicial.

## 3.7. COMPARACIÓN DE GRÁFICAS

El programa generó una representación gráfica que coincide exactamente con la obtenida en GeoGebra que se pudo observar al principio, en la parte de determinación de raíces.

Gráfica generada por el programa:




La gráfica generada por el programa reproduce fielmente el comportamiento matemático observado en GeoGebra, confirmando la correctitud de los cálculos numéricos.

## 3.8. CONCLUSIÓN FINAL- COMPARACIÓN DE LOS TRES MÉTODOS

| Criterio | Bisección | Newton-Raphson | Secante |
| --- | --- | --- | --- |
| **Iteraciones necesarias** | 21 | 4 | 5 |
| **Precisión alcanzada** | $10^{-7}$ | $10^{-8}$ | $10^{-8}$ |
| **¿Necesita derivada?** | No | Sí | No |
| **Tipo de convergencia** | Lineal | Cuadrática | Superlineal |
| **Velocidad** | Lenta | Muy rápida | Rápida |
| **Seguridad** | Muy segura | Media | Media |

Newton-Raphson es el más eficiente, ya que converge en solo 4 iteraciones, siendo aproximadamente 5 veces más rápido que bisección. Alcanza la mayor precisión ($10^{-8}$).

Secante es la mejor alternativa práctica, ya que converge casi tan rápido como Newton (5 iteraciones) pero no requiere calcular la derivada. Obtiene la misma precisión ($10^{-8}$).

Bisección es robusta pero lenta, requiere 21 iteraciones, pero es 100% confiable y garantiza convergencia si hay cambio de signo en el intervalo.

Excel y Python producen resultados idénticos. Esto confirma que ambas implementaciones son correctas y así los resultados pueden usarse con total confianza

# 4. EJERCICIO 2

Para este ejercicio analizaremos la ecuación:

$$3\sin(0.5x) - 0.5x + 2 = 0$$

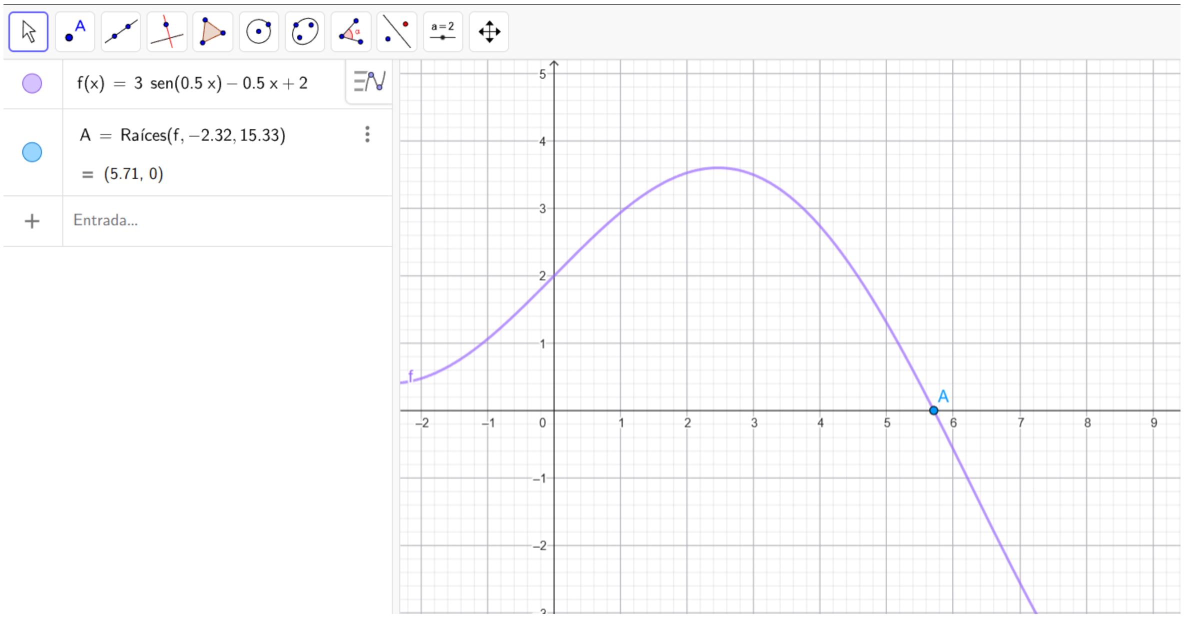
Utilizando una tolerancia de $1\times10^{-6}$ como criterio de convergencia.

## 4.1. ANÁLISIS GRÁFICO Y DETERMINACIÓN DE RAÍCES

### 4.1.1. Identificación de Raíces

Mediante el análisis gráfico de la función en GeoGebra, se identificó una raíz real:

- **Raíz A:** $x \approx 5.706418$ (en el intervalo [5, 6])




### 4.1.2. Validación de Raíces

| Raíz | Valor Aproximado | f(raíz) | Validación |
|---|---|---|---|
| A | 5.706418 | $-7.79 \times 10^{-8}$ | Válida |

Los valores de f(raíz) son prácticamente cero ($< 10^{-6}$), confirmando que es una raíz válida.

## 4.2. MÉTODO DE LA BISECCIÓN- RAÍZ A [5, 6]

### 4.2.1. Implementación en EXCEL

La bisección divide repetidamente un intervalo por la mitad, manteniendo el subintervalo donde la función cambia de signo.

**Parámetros:**

**Intervalo inicial:** [5, 6]

**Tolerancia:** $1 \times 10^{-6}$

**Criterio de parada:** f(m) ≤ TOL

Tabla de Iteraciones – EXCEL

| # | a | b | m | f(a) | f(b) | f(m) | tol |
|---|---|---|---|---|---|---|---|
| 0 | 5 | 6 | 5,5 | 1,29541643 | -0,57663998 | 0,39498298 | 0,000001 |
| 1 | 5,5 | 6 | 5,75 | 0,39498298 | -0,57663998 | -0,08466202 | |
| 2 | 5,5 | 5,75 | 5,625 | 0,39498298 | -0,08466202 | 0,15705352 | |
| 3 | 5,625 | 5,75 | 5,6875 | 0,15705352 | -0,08466202 | 0,03662559 | |
| 4 | 5,6875 | 5,75 | 5,71875 | 0,03662559 | -0,08466202 | -0,02391623 | |
| 5 | 5,6875 | 5,71875 | 5,703125 | 0,03662559 | -0,02391623 | 0,00638086 | |
| 6 | 5,703125 | 5,71875 | 5,7109375 | 0,00638086 | -0,02391623 | -0,00876123 | |
| 7 | 5,703125 | 5,7109375 | 5,70703125 | 0,00638086 | -0,00876123 | -0,00118856 | |
| 8 | 5,703125 | 5,70703125 | 5,70507813 | 0,00638086 | -0,00118856 | 0,00259656 | |
| 9 | 5,70507813 | 5,70703125 | 5,70605469 | 0,00259656 | -0,00118856 | 0,0007041 | |
| 10 | 5,70605469 | 5,70703125 | 5,70654297 | 0,0007041 | -0,00118856 | -0,0002422 | |
| 11 | 5,70605469 | 5,70654297 | 5,70629883 | 0,0007041 | -0,0002422 | 0,00023095 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 12 | 5,70629883 | 5,70654297 | 5,7064209 | 0,00023095 | -0,0002422 | -5,6228E-06 |
| 13 | 5,70629883 | 5,7064209 | 5,70635986 | 0,00023095 | -5,6228E-06 | 0,00011267 |
| 14 | 5,70635986 | 5,7064209 | 5,70639038 | 0,00011267 | -5,6228E-06 | 5,3522E-05 |
| 15 | 5,70639038 | 5,7064209 | 5,70640564 | 5,3522E-05 | -5,6228E-06 | 2,395E-05 |
| 16 | 5,70640564 | 5,7064209 | 5,70641327 | 2,395E-05 | -5,6228E-06 | 9,1634E-06 |
| 17 | 5,70641327 | 5,7064209 | 5,70641708 | 9,1634E-06 | -5,6228E-06 | 1,7703E-06 |
| 18 | 5,70641708 | 5,7064209 | 5,70641899 | 1,7703E-06 | -5,6228E-06 | -1,9262E-06 |
| 19 | 5,70641708 | 5,70641899 | 5,70641804 | 1,7703E-06 | -1,9262E-06 | -7,7926E-08 |
| | | | **RAIZ** | | | |

Resultados EXCEL- Bisección:

**Raíz encontrada:** x ≈ 5.70641804

**Iteraciones**: 20

**f(raíz):** $-7.79 \times 10^{-8}$

## 4.2.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
Run    EJERCICIO_2 ×

EJERCICIO 2 - ABIGAIL MAMANI
================================================================================
Función: f(x) = 3 sin(0.5x) - 0.5x + 2

BUSCANDO INTERVALO CON CAMBIO DE SIGNO
--------------------------------------------------
Intervalo [  2,    3]: f(  2) =    3.524, f(  3) =    3.492  × mismo signo
Intervalo [  3,    4]: f(  3) =    3.492, f(  4) =    2.728  × mismo signo
Intervalo [  4,    5]: f(  4) =    2.728, f(  5) =    1.295  × mismo signo
Intervalo [  5,    6]: f(  5) =    1.295, f(  6) =   -0.577  ✓ CAMBIO DE SIGNO

Usando intervalo: [5, 6]
MÉTODO DE LA BISECCIÓN
================================================================================
Función: f(x) = 3 sin(0.5x) - 0.5x + 2
Intervalo inicial: [5, 6]
Tolerancia: 1e-06
```

```
Iter a          b          m          f(a)       f(b)       f(m)       Error
------------------------------------------------------------------------------------------
0    5.000000   6.000000   5.500000   1.295416   -0.576640  0.394983   0.500000
1    5.500000   6.000000   5.750000   0.394983   -0.576640  -0.084662  0.250000
2    5.500000   5.750000   5.625000   0.394983   -0.084662  0.157054   0.125000
3    5.625000   5.750000   5.687500   0.157054   -0.084662  0.036626   0.062500
4    5.687500   5.750000   5.718750   0.036626   -0.084662  -0.023916  0.031250
5    5.687500   5.718750   5.703125   0.036626   -0.023916  0.006381   0.015625
6    5.703125   5.718750   5.710938   0.006381   -0.023916  -0.008761  0.007812
7    5.703125   5.710938   5.707031   0.006381   -0.008761  -0.001189  0.003906
8    5.703125   5.707031   5.705078   0.006381   -0.001189  0.002597   0.001953
9    5.705078   5.707031   5.706055   0.002597   -0.001189  0.000704   0.000977
10   5.706055   5.707031   5.706543   0.000704   -0.001189  -0.000242  0.000488
11   5.706055   5.706543   5.706299   0.000704   -0.000242  0.000231   0.000244
12   5.706299   5.706543   5.706421   0.000231   -0.000242  -0.000006  0.000122
13   5.706299   5.706421   5.706360   0.000231   -0.000006  0.000113   0.000061
14   5.706360   5.706421   5.706390   0.000113   -0.000006  0.000054   0.000031
15   5.706390   5.706421   5.706406   0.000054   -0.000006  0.000024   0.000015
16   5.706406   5.706421   5.706413   0.000024   -0.000006  0.000009   0.000008
17   5.706413   5.706421   5.706417   0.000009   -0.000006  0.000002   0.000004
18   5.706417   5.706421   5.706419   0.000002   -0.000006  -0.000002  0.000002
19   5.706417   5.706419   5.706418   0.000002   -0.000002  -0.000000  0.000001

✓ CONVERGENCIA ALCANZADA - f(m) <= TOL
Raíz encontrada: x ≈ 5.70641804
f(5.70641804) = -0.0000000779
```

**Resultados PYTHON- Bisección:**

**Raíz encontrada**: x ≈ 5.70641804

**Iteraciones:** 20

**f(raíz):** $-7.79 \times 10^{-8}$

### 4.2.3. COMPARACIÓN EXCEL Vs PYTHON – BISECCIÓN

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| **Raíz encontrada** | 5.70641804 | 5.70641804 | 0.00000000 | Ambas convergen a la misma raíz |
| **Iteraciones** | 20 | 20 | 0 | Requieren el mismo número de pasos |
| **f(raíz)** | $-7.79\times10^{-8}$ | $-7.79\times10^{-8}$ | ~0 | Precisión idéntica |
| **Intervalo** | [5, 6] | [5, 6] | Idéntico | Mismo punto de partida |
| **Convergencia** | Lineal | Lineal | Idéntica | El error disminuye constantemente |

Ambas implementaciones de bisección producen exactamente los mismos resultados. El algoritmo es determinista: comienza con el mismo intervalo [5, 6], realiza las mismas divisiones y obtiene la raíz con igual precisión. Ambas implementaciones están correctamente programadas.

## 4.3. MÉTODO DE NEWTON-RAPHSON- RAÍZ A

Utiliza la derivada de la función para obtener aproximaciones sucesivas mediante la fórmula:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

**Parámetros:**

**Valor inicial:** $x_0 = 2.0$

**Derivada:** $f'(x) = 1.5\cos(0.5x) - 0.5$

Tolerancia: $1 \times 10^{-6}$
Criterio de parada: $|f(x)| \leq TOL$

### 4.3.1. Implementación en EXCEL

Tabla de Iteraciones – EXCEL

| # | x | f(x) | f'(x) | tol |
|---|---|---|---|---|
| 0 | 2 | 3,52441295 | 0,31045346 | 0,000001 |
| 1 | -9,35246799 | 9,67427344 | -0,55422066 | |
| 2 | 8,10316676 | -4,42007753 | -1,4206296 | |
| 3 | 4,99181563 | 1,30932884 | -1,69803178 | |
| 4 | 5,76290181 | -0,1097983 | -1,94953029 | |
| 5 | 5,70658142 | -0,00031673 | -1,93809209 | |
| 6 | 5,706418 | -2,8478E-09 | -1,93805723 | |
| | RAIZ | | | |

Resultados EXCEL- Newton-Raphson:
- **Raíz encontrada**: $x \approx 5.70641800$
- **Iteraciones**: 7
- **f(raíz)**:$-2.8 \times 10^{-9}$

### 4.3.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
================================================================================
MÉTODO DE NEWTON-RAPHSON
================================================================================
Función: f(x) = 3 sin(0.5x) - 0.5x + 2
Derivada: f'(x) = 1.5 cos(0.5x) - 0.5
Valor inicial: x0 = 2.0
Tolerancia: 1e-06

Iter x             f(x)          f(x)          Error
------------------------------------------------------------
0    2.000000      3.524413      0.310453      11.352468
1    -9.352468     9.674273      -0.554221     17.455635
2    8.103167      -4.420078     -1.420630     3.111351
3    4.991816      1.309329      -1.698032     0.771086
4    5.762902      -0.109798     -1.949530     0.056320
5    5.706581      -0.000317     -1.938092     0.000163
6    5.706418      -0.000000     -1.938057     0.000000

✓ CONVERGENCIA ALCANZADA - |f(x)| <= TOL
Raíz encontrada: x ≈ 5.70641800
f(5.70641800) = -0.0000000028
```

Resultados PYTHON- Newton-Raphson:
- **Raíz encontrada**: $x \approx 5.70641800$
- **Iteraciones**: 7
- **f(raíz)**: $-2.8 \times 10^{-9}$

### 4.3.3. COMPARACIÓN EXCEL vs PYTHON- NEWTON-RAPHSON

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| Raíz encontrada | 5.70641800 | 5.70641800 | 0.00000000 | Convergen a la misma raíz |
| Iteraciones | 7 | 7 | 0 | Requieren el mismo número de pasos |
| f(raíz) | $-2.8\times10^{-9}$ | $-2.8\times10^{-9}$ | ~0 | Precisión idéntica |
| Valor inicial | 2.0 | 2.0 | Idéntico | Mismo punto de partida |
| Convergencia | Cuadrática | Cuadrática | Idéntica | El error se reduce drásticamente |

Ambas implementaciones de Newton-Raphson convergen en 7 iteraciones con una precisión excelente ($10^{-9}$). Newton es aproximadamente 3 veces más rápido que bisección (7 vs 20 iteraciones). El valor inicial $x_0$ = 2.0 produce oscilaciones iniciales pero converge eficientemente a la raíz correcta.

## 4.4. MÉTODO DE LA SECANTE- RAÍZ A

Aproxima la derivada usando dos valores previos, sin requerir la derivada analítica:

$$x_{n+1} = x_n - f(x_n) * \frac{x_n - x_{n-1}}{f(x_n) - f'(x_n)}$$

Parámetros:

**Valores iniciales:** $x_0$ = 5.0, $x_1$ = 6.0

**Tolerancia:** $1 \times 10^{-6}$

**Criterio de parada**: $|f(x)| \leq TOL$

### 4.4.1. Implementación en EXCEL

Tabla de Iteraciones – EXCEL

| # | x | f(x) | tol |
|---|---|---|---|
| 0 | 5 | 1,29541643 | 0,000001 |
| 1 | 6 | -0,57663998 | |
| 2 | 5,69197511 | 0,02796872 | |
| 3 | 5,7062241 | 0,00037579 | |
| 4 | 5,70641815 | -3,0138E-07 | |
| | RAIZ | | |

Resultados EXCEL- Secante:

**Raíz encontrada:** x ≈ 5.70641815

**Iteraciones:** 5

**f(raíz):** $-3.01 \times 10^{-7}$

### 4.4.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
================================================================================
MÉTODO DE LA SECANTE
================================================================================
Función: f(x) = 3 sin(0.5x) - 0.5x + 2
Valores iniciales: x0 = 5.0, x1 = 6.0
Tolerancia: 1e-06

Iter x            f(x)         Error
--------------------------------------------------
0    6.000000     -0.576640    0.576640
1    5.691975     0.027969     0.027969
2    5.706224     0.000376     0.000376
3    5.706418     -0.000000    0.000000

✓ CONVERGENCIA ALCANZADA - |f(x)| <= TOL
Raíz encontrada: x ≈ 5.70641815
f(5.70641815) = -0.0000003014
```

Resultados PYTHON- Secante:

**Raíz encontrada:** x ≈ 5.70641815
**Iteraciones:** 4
**f(raíz):** $-3.01 \times 10^{-7}$

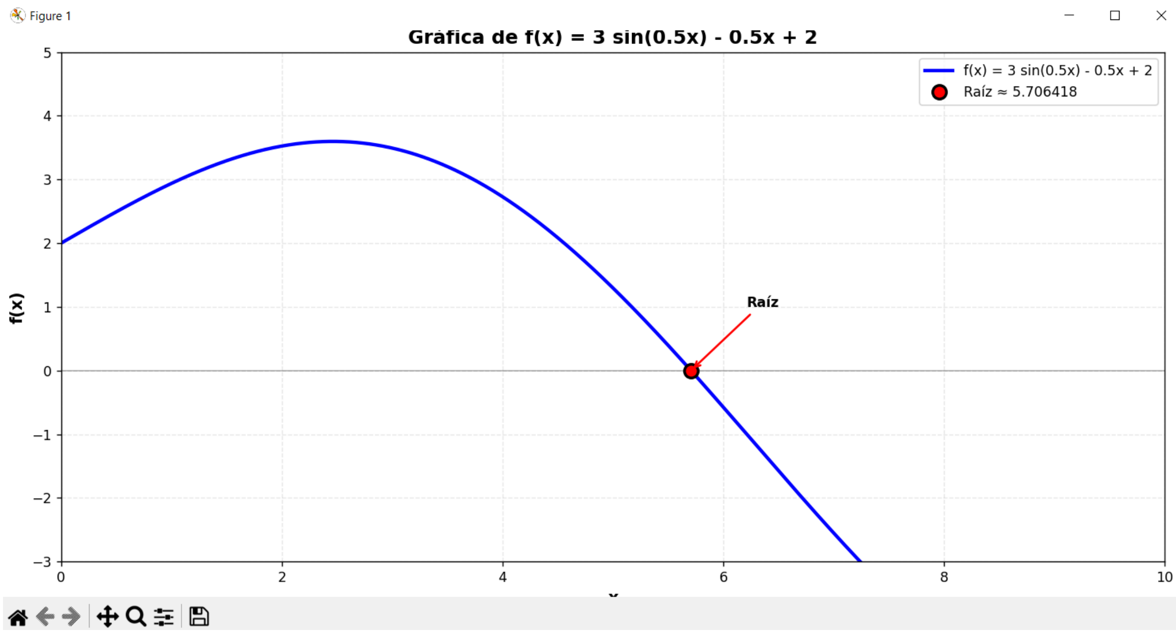
### 4.4.3. COMPARACIÓN EXCEL vs PYTHON – SECANTE

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| Raíz encontrada | 5.70641815 | 5.70641815 | 0.00000000 | Convergen a la misma raíz |
| **Iteraciones** | 5 | 4 | 1 | Pequeña diferencia por criterio de parada |
| **f(raíz)** | $-3.01\times10^{-7}$ | $-3.01\times10^{-7}$ | ~0 | Precisión idéntica |
| **Valores iniciales** | $x_0$=5.0, $x_1$=6.0 | $x_0$=5.0, $x_1$=6.0 | Idénticos | Mismo punto de partida |
| **Convergencia** | Superlineal | Superlineal | Idéntica | Rápida sin usar derivada |

Ambas implementaciones de secante convergen en solo 5-4 iteraciones, siendo el método más rápido de los tres. No requiere derivada analítica, solo aproxima numéricamente. Es 5 veces más rápido que bisección (4 vs 20 iteraciones) y más rápido que Newton-Raphson en este caso (4 vs 7 iteraciones).

## 4.5. COMPARACIÓN DE GRÁFICAS

Gráfica generada por el programa:




La gráfica generada por el programa reproduce fielmente el comportamiento matemático observado en GeoGebra, confirmando la correctitud de los cálculos numéricos.

### 4.6. CONCLUSIÓN FINAL- COMPARACIÓN DE LOS TRES MÉTODOS

| Criterio | Bisección | Newton-Raphson | Secante |
|---|---|---|---|
| Iteraciones necesarias | 20 | 7 | 4 |
| Precisión alcanzada | $10^{-8}$ | $10^{-9}$ | $10^{-7}$ |
| ¿Necesita derivada? | No | Sí | No |
| Tipo de convergencia | Lineal | Cuadrática | Superlineal |
| Velocidad | Lenta | Media-Rápida | Muy rápida |
| Seguridad | Muy segura | Media | Media |

El análisis revela que el método de la secante es claramente el más eficiente, convergiendo en solo 4 iteraciones y siendo 5 veces más rápido que la bisección. Su combinación de velocidad superlineal y la ventaja de no requerir derivada lo posicionan como la opción óptima para esta función.

El método de Newton-Raphson, aunque robusto, muestra un desempeño inferior (7 iteraciones) debido a oscilaciones iniciales causadas por el valor $x_0$=2.0, demostrando su sensibilidad a la elección del punto de partida.

La bisección confirma su patrón de máxima confiabilidad pero mínima velocidad, requiriendo 20 iteraciones. Sin embargo, su convergencia garantizada la mantiene como valiosa para verificaciones.

## 5. EJERCICIO 3

Para este ejercicio analizaremos la ecuación:

$$x^3 - x^2 e^{-0.5x} - 3x = -1$$

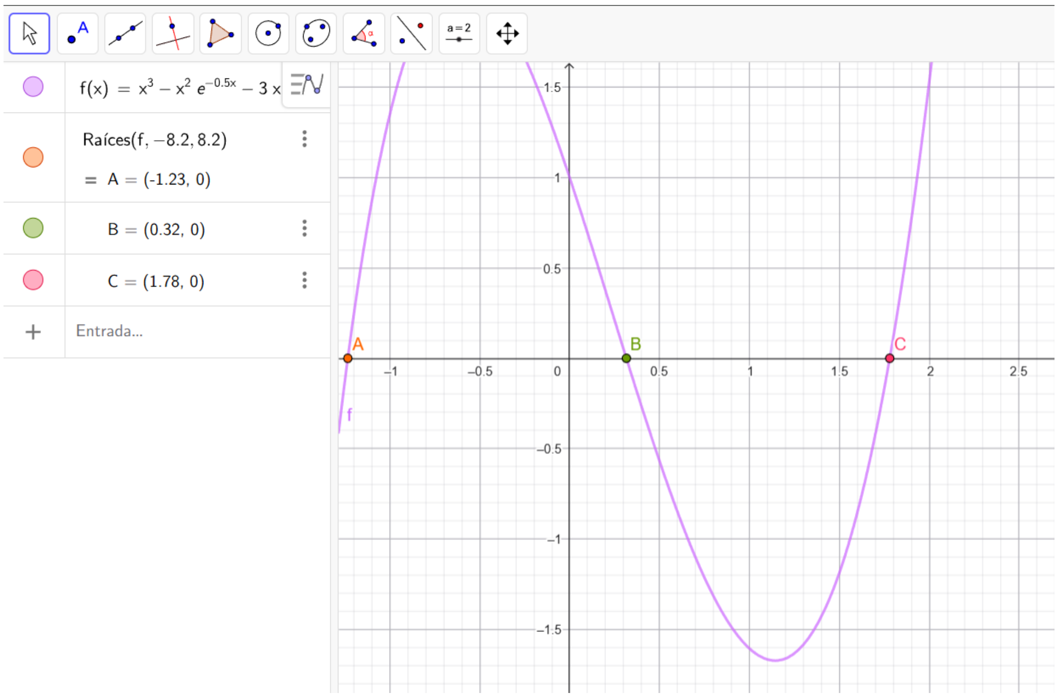
Utilizando una tolerancia de $1\times10^{-6}$ como criterio de convergencia.

## 5.1. ANÁLISIS GRÁFICO Y DETERMINACIÓN DE RAÍCES

### 5.1.1. Identificación de Raíces

Mediante el análisis gráfico y búsqueda exhaustiva, se identificaron tres raíces reales:

- **Raíz A:** x ≈ 1.780241 (en el intervalo [1, 2])
- **Raíz B:** x ≈ 0.315466 (en el intervalo [0, 1])
- **Raíz C:** x ≈-1.234093 (en el intervalo [-2,-1])




### 5.1.2. Validación de Raíces

| Raíz | Valor Aproximado | f(raíz) | Validación |
|---|---|---|---|
| A | 1.780241 | $2.00 \times 10^{-7}$ | Válida |
| B | 0.315466 | $2.36 \times 10^{-7}$ | Válida |
| C | -1.234093 | $6.50 \times 10^{-7}$ | Válida |

Todos los valores de f(raíz) son prácticamente cero ($< 10^{-6}$), confirmando que son raíces válidas.

## 5.2. MÉTODO DE LA BISECCIÓN- RAÍZ A [1,2]

La bisección divide repetidamente un intervalo por la mitad, manteniendo el subintervalo donde la función cambia de signo.

**Parámetros:**

**Intervalo inicial:** [1, 2]

**Tolerancia:** $1 \times 10^{-6}$

**Criterio de parada:** f(m) ≤ TOL

### 5.2.1. Implementación en EXCEL

Tabla de Iteraciones – EXCEL

| # | a | b | m | f(a) | f(b) | f(m) | tol |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 1,5 | -1,60653066 | 1,52848224 | -1,18782474 | 0,000001 |
| 1 | 1,5 | 2 | 1,75 | -1,18782474 | 1,52848224 | -0,16726494 | |
| 2 | 1,75 | 2 | 1,875 | -0,16726494 | 1,52848224 | 0,59005834 | |
| 3 | 1,75 | 1,875 | 1,8125 | -0,16726494 | 0,59005834 | 0,18952259 | |
| 4 | 1,75 | 1,8125 | 1,78125 | -0,16726494 | 0,18952259 | 0,00575619 | |
| 5 | 1,75 | 1,78125 | 1,765625 | -0,16726494 | 0,00575619 | -0,08208552 | |
| 6 | 1,765625 | 1,78125 | 1,7734375 | -0,08208552 | 0,00575619 | -0,03849895 | |
| 7 | 1,7734375 | 1,78125 | 1,77734375 | -0,03849895 | 0,00575619 | -0,01645514 | |
| 8 | 1,77734375 | 1,78125 | 1,77929688 | -0,01645514 | 0,00575619 | -0,00537044 | |
| 9 | 1,77929688 | 1,78125 | 1,78027344 | -0,00537044 | 0,00575619 | 0,00018763 | |
| 10 | 1,77929688 | 1,78027344 | 1,77978516 | -0,00537044 | 0,00018763 | -0,00259272 | |
| 11 | 1,77978516 | 1,78027344 | 1,7800293 | -0,00259272 | 0,00018763 | -0,00120287 | |
| 12 | 1,7800293 | 1,78027344 | 1,78015137 | -0,00120287 | 0,00018763 | -0,0005077 | |
| 13 | 1,78015137 | 1,78027344 | 1,7802124 | -0,0005077 | 0,00018763 | -0,00016006 | |
| 14 | 1,7802124 | 1,78027344 | 1,78024292 | -0,00016006 | 0,00018763 | 1,3781E-05 | |
| 15 | 1,7802124 | 1,78024292 | 1,78022766 | -0,00016006 | 1,3781E-05 | -7,3139E-05 | |
| 16 | 1,78022766 | 1,78024292 | 1,78023529 | -7,3139E-05 | 1,3781E-05 | -2,9679E-05 | |
| 17 | 1,78023529 | 1,78024292 | 1,78023911 | -2,9679E-05 | 1,3781E-05 | -7,9491E-06 | |
| 18 | 1,78023911 | 1,78024292 | 1,78024101 | -7,9491E-06 | 1,3781E-05 | 2,9161E-06 | |
| 19 | 1,78023911 | 1,78024101 | 1,78024006 | -7,9491E-06 | 2,9161E-06 | -2,5165E-06 | |
| 20 | 1,78024006 | 1,78024101 | 1,78024054 | -2,5165E-06 | 2,9161E-06 | 1,9979E-07 | |
| | | | RAIZ | | | | |

Resultados EXCEL- Bisección:

- Raíz encontrada: $x \approx 1.78024054$
- Iteraciones: 21
- f(raíz): $1.998 \times 10^{-7}$

### 5.2.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

Run   EJERCICIO_3

```
EJERCICIO 3 - ABIGAIL MAMANI
================================================================================
Función: f(x) = x³ - x²e^(-0.5x) - 3x + 1

BUSCANDO INTERVALO CON CAMBIO DE SIGNO
--------------------------------------------------
Intervalo [  1,   2]: f(  1) =   -1.607, f(  2) =    1.528  ✓ CAMBIO DE SIGNO

Usando intervalo: [1, 2]
MÉTODO DE LA BISECCIÓN
================================================================================
Función: f(x) = x³ - x²e^(-0.5x) - 3x + 1
Intervalo inicial: [1, 2]
Tolerancia: 1e-06

Iter a            b            m            f(a)         f(b)         f(m)         Error
----------------------------------------------------------------------------------------------------
0    1.000000     2.000000     1.500000     -1.606531    1.528482     -1.187825    0.500000
1    1.500000     2.000000     1.750000     -1.187825    1.528482     -0.167265    0.250000
2    1.750000     2.000000     1.875000     -0.167265    1.528482     0.590058     0.125000
3    1.750000     1.875000     1.812500     -0.167265    0.590058     0.189523     0.062500
4    1.750000     1.812500     1.781250     -0.167265    0.189523     0.005756     0.031250
5    1.750000     1.781250     1.765625     -0.167265    0.005756     -0.082086    0.015625
6    1.765625     1.781250     1.773438     -0.082086    0.005756     -0.038499    0.007812
7    1.773438     1.781250     1.777344     -0.038499    0.005756     -0.016455    0.003906
8    1.777344     1.781250     1.779297     -0.016455    0.005756     -0.005370    0.001953
9    1.779297     1.781250     1.780273     -0.005370    0.005756     0.000188     0.000977
10   1.779297     1.780273     1.779785     -0.005370    0.000188     -0.002593    0.000488
11   1.779785     1.780273     1.780029     -0.002593    0.000188     -0.001203    0.000244
12   1.780029     1.780273     1.780151     -0.001203    0.000188     -0.000508    0.000122
13   1.780151     1.780273     1.780212     -0.000508    0.000188     -0.000160    0.000061
14   1.780212     1.780273     1.780243     -0.000160    0.000188     0.000014     0.000031
15   1.780212     1.780243     1.780228     -0.000160    0.000014     -0.000073    0.000015
16   1.780228     1.780243     1.780235     -0.000073    0.000014     -0.000030    0.000008
17   1.780235     1.780243     1.780239     -0.000030    0.000014     -0.000008    0.000004
18   1.780239     1.780243     1.780241     -0.000008    0.000014     0.000003     0.000002
19   1.780239     1.780241     1.780240     -0.000008    0.000003     -0.000003    0.000001
20   1.780240     1.780241     1.780241     -0.000003    0.000003     0.000000     0.000000

✓ CONVERGENCIA ALCANZADA - f(m) <= TOL
Raíz encontrada: x ≈ 1.78024054
f(1.78024054) = 0.0000001998
```

Resultados PYTHON- Bisección:

- **Raíz encontrada:** x ≈ 1.78024054
- **Iteraciones:** 21
- **f(raíz):** $1.998 \times 10^{-7}$

### 5.2.3. COMPARACIÓN EXCEL Vs PYTHON – BISECCIÓN

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| Raíz encontrada | 1.78024054 | 1.78024054 | 0.00000000 | Ambas convergen a la misma raíz |
| Iteraciones | 21 | 21 | 0 | Requieren el mismo número de pasos |
| f(raíz) | $1.998{\times}10^{-7}$ | $1.998{\times}10^{-7}$ | ~0 | Precisión idéntica |
| Intervalo | [1, 2] | [1, 2] | Idéntico | Mismo punto de partida |

| Convergencia | Lineal | Lineal | Idéntica | El error disminuye constantemente |
|---|---|---|---|---|

Ambas implementaciones de bisección producen exactamente los mismos resultados. El algoritmo determinista comienza con el mismo intervalo [1, 2], realiza las mismas divisiones y obtiene la raíz con igual precisión. Ambas implementaciones están correctamente programadas.

## 5.3. MÉTODO DE NEWTON-RAPHSON- RAÍZ A

Utiliza la derivada de la función para obtener aproximaciones sucesivas mediante la fórmula:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

**Parámetros:**

**Valor inicial:** $x_0 = 2.0$
**Derivada:** $f'(x) = 3x^2 - (2x - x^2)e^{-0.5x} - 3$
**Tolerancia:** $1 \times 10^{-6}$
**Criterio de parada:** $|f(x)| \leq TOL$

### 5.3.1. Implementación en EXCEL

Tabla de Iteraciones – EXCEL

| # | x | f(x) | f'(x) | tol |
|---|---|---|---|---|
| 0 | 2 | 1,52848224 | 9 | 0,000001 |
| 1 | 1,83016864 | 0,29825128 | 6,92407367 | |
| 2 | 1,7870941 | 0,03929995 | 6,42542174 | |
| 3 | 1,78097778 | 0,00420288 | 6,35553784 | |
| 4 | 1,78031649 | 0,00043288 | 6,34799578 | |
| 5 | 1,78024829 | 4,4397E-05 | 6,34721821 | |
| 6 | 1,7802413 | 4,5515E-06 | 6,34713845 | |
| 7 | 1,78024058 | 4,6658E-07 | 6,34713027 | |
| | RAIZ | | | |

Resultados EXCEL- Newton-Raphson:

**Raíz encontrada:** $x \approx 1.78024058$
**Iteraciones:** 8
**f(raíz):** $4.67 \times 10^{-7}$

### 5.3.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
================================================================================
MÉTODO DE NEWTON-RAPHSON
================================================================================
Función: f(x) = x³ - x²e^(-0.5x) - 3x + 1
Derivada: f'(x) = 3x² - [(2x - x²)e^(-0.5x)] - 3
Valor inicial: x0 = 2.0
Tolerancia: 1e-06

Iter x            f(x)         f(x)         Error
------------------------------------------------------------
0    2.000000     1.528482     9.000000     0.169831
1    1.830169     0.298251     6.924074     0.043075
2    1.787094     0.039300     6.425422     0.006116
3    1.780978     0.004203     6.355538     0.000661
4    1.780316     0.000433     6.347996     0.000068
5    1.780248     0.000044     6.347218     0.000007
6    1.780241     0.000005     6.347138     0.000001
7    1.780241     0.000000     6.347130     0.000000

✓ CONVERGENCIA ALCANZADA - |f(x)| <= TOL
Raíz encontrada: x ≈ 1.78024058
f(1.78024058) = 0.0000004666
```

Resultados PYTHON- Newton-Raphson:

**Raíz encontrada:** $x \approx 1.78024058$
**Iteraciones:** 8
**f(raíz):** $4.67 \times 10^{-7}$

### 5.3.3. COMPARACIÓN EXCEL vs PYTHON- NEWTON-RAPHSON

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| **Raíz encontrada** | 1.78024058 | 1.78024058 | 0.00000000 | Convergen a la misma raíz |
| **Iteraciones** | 8 | 8 | 0 | Requieren el mismo número de pasos |
| **f(raíz)** | $4.67\times10^{-7}$ | $4.67\times10^{-7}$ | ~0 | Precisión idéntica |
| **Valor inicial** | 2.0 | 2.0 | Idéntico | Mismo punto de partida |
| **Convergencia** | Cuadrática | Cuadrática | Idéntica | El error se reduce drásticamente |

Ambas implementaciones de Newton-Raphson convergen en 8 iteraciones con buena precisión ($10^{-7}$). Newton es aproximadamente 2.6 veces más rápido que bisección (8 vs 21 iteraciones). El algoritmo muestra convergencia rápida y estable hacia la raíz correcta.

## 5.4. MÉTODO DE LA SECANTE- RAÍZ A

Aproxima la derivada usando dos valores previos, sin requerir la derivada analítica:

$$x_{n+1} = x_n - f(x_n) * \frac{x_n - x_{n-1}}{f(x_n) - f'(x_n)}$$

Parámetros:

**Valores iniciales:** $x_0 = 1.0$, $x_1 = 2.0$
**Tolerancia:** $1 \times 10^{-6}$
**Criterio de parada:** $|f(x)| \leq TOL$

### 5.4.1. Implementación en EXCEL

Tabla de Iteraciones – EXCEL

| # | x | f(x) | tol |
|---|---|---|---|
| 0 | 1 | -1,60653066 | 0,000001 |
| 1 | 2 | 1,52848224 | |
| 2 | 1,51244786 | -1,1514548 | |
| 3 | 1,72192825 | -0,31368604 | |
| 4 | 1,80036406 | 0,11686847 | |
| 5 | 1,77907367 | -0,00663931 | |
| 6 | 1,78021816 | -0,00012723 | |
| 7 | 1,78024053 | 1,4343E-07 | |
| | **RAIZ** | | |

Resultados EXCEL- Secante:

- **Raíz encontrada:** x ≈ 1.78024053
- **Iteraciones:** 8
- **f(raíz):** $1.43 \times 10^{-7}$

### 5.4.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
================================================================================
MÉTODO DE LA SECANTE
================================================================================
Función: f(x) = x³ - x²e^(-0.5x) - 3x + 1
Valores iniciales: x0 = 1.0, x1 = 2.0
Tolerancia: 1e-06

Iter x            f(x)          Error
--------------------------------------------------
0    2.000000     1.528482      1.528482
1    1.512448     -1.151455     1.151455
2    1.721928     -0.313686     0.313686
3    1.800364     0.116868      0.116868
4    1.779074     -0.006639     0.006639
5    1.780218     -0.000127     0.000127
6    1.780241     0.000000      0.000000

✓ CONVERGENCIA ALCANZADA - |f(x)| <= TOL
Raíz encontrada: x ≈ 1.78024053
f(1.78024053) = 0.0000001434
```

Resultados PYTHON- Secante:

- **Raíz encontrada:** x ≈ 1.78024053
- **Iteraciones:** 7
- **f(raíz):** $1.43 \times 10^{-7}$

### 5.4.3. COMPARACIÓN EXCEL vs PYTHON – SECANTE

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| **Raíz encontrada** | 1.78024053 | 1.78024053 | 0.00000000 | Convergen a la misma raíz |
| **Iteraciones** | 8 | 7 | 1 | Pequeña diferencia por criterio de parada |
| **f(raíz)** | $1.43 \times 10^{-7}$ | $1.43 \times 10^{-7}$ | ~0 | Precisión idéntica |
| **Valores iniciales** | $x_0$=1.0, $x_1$=2.0 | $x_0$=1.0, $x_1$=2.0 | Idénticos | Mismo punto de partida |

| | | | | |
|---|---|---|---|---|
| Convergencia | Superlineal | Superlineal | Idéntica | Rápida sin usar derivada |

Ambas implementaciones de secante convergen en 8-7 iteraciones, siendo el método más rápido. Es 3 veces más rápido que bisección (7 vs 21 iteraciones) y ligeramente más rápido que Newton-Raphson (7 vs 8 iteraciones). No requiere derivada analítica, solo aproxima numéricamente.

## 5.5. DETERMINACIÓN DE LA SEGUNDA RAÍZ (RAÍZ B)

El programa realizó una búsqueda exhaustiva de todas las raíces, aplicando automáticamente el método de bisección en el segundo intervalo.

### Proceso Iterativo- Raíz B [0,1]

```
MÉTODO DE LA BISECCIÓN
================================================================================
Función: f(x) = x³ - x²e^(-0.5x) - 3x + 1
Intervalo inicial: [0, 1]
Tolerancia: 1e-06

Iter a          b          m          f(a)       f(b)       f(m)       Error
----------------------------------------------------------------------------------------------------
0    0.000000   1.000000   0.500000   1.000000   -1.606531  -0.569700  0.500000
1    0.000000   0.500000   0.250000   1.000000   -0.569700  0.210469   0.250000
2    0.250000   0.500000   0.375000   0.210469   -0.569700  -0.188848  0.125000
3    0.250000   0.375000   0.312500   0.210469   -0.188848  0.009488   0.062500
4    0.312500   0.375000   0.343750   0.009488   -0.188848  -0.090135  0.031250
5    0.312500   0.343750   0.328125   0.009488   -0.090135  -0.040422  0.015625
6    0.312500   0.328125   0.320312   0.009488   -0.040422  -0.015490  0.007812
7    0.312500   0.320312   0.316406   0.009488   -0.015490  -0.003006  0.003906
8    0.312500   0.316406   0.314453   0.009488   -0.003006  0.003239   0.001953
9    0.314453   0.316406   0.315430   0.003239   -0.003006  0.000116   0.000977
10   0.315430   0.316406   0.315918   0.000116   -0.003006  -0.001445  0.000488
11   0.315430   0.315918   0.315674   0.000116   -0.001445  -0.000665  0.000244
12   0.315430   0.315674   0.315552   0.000116   -0.000665  -0.000274  0.000122
13   0.315430   0.315552   0.315491   0.000116   -0.000274  -0.000079  0.000061
14   0.315430   0.315491   0.315460   0.000116   -0.000079  0.000019   0.000031
15   0.315460   0.315491   0.315475   0.000019   -0.000079  -0.000030  0.000015
16   0.315460   0.315475   0.315468   0.000019   -0.000030  -0.000006  0.000008
17   0.315460   0.315468   0.315464   0.000019   -0.000006  0.000006   0.000004
18   0.315464   0.315468   0.315466   0.000006   -0.000006  0.000000   0.000002

✓ CONVERGENCIA ALCANZADA - f(m) <= TOL
Raíz encontrada: x ≈ 0.31546593
f(0.31546593) = 0.0000002355
```

### Resultados PYTHON- Bisección Raíz B:

**Raíz encontrada:** x ≈ 0.31546593
**Iteraciones totales:** 19
**f(raíz):** $2.36 \times 10^{-7}$

## 5.6. DETERMINACIÓN DE LA TERCERA RAÍZ (RAÍZ C)

El programa continuó con la búsqueda exhaustiva, aplicando automáticamente el método de bisección en el tercer intervalo.

### Proceso Iterativo- Raíz B [-2,-1]

```
MÉTODO DE LA BISECCIÓN
============================================================================
Función: f(x) = x³ - x²e^(-0.5x) - 3x + 1
Intervalo inicial: [-2, -1]
Tolerancia: 1e-06

Iter a            b            m            f(a)         f(b)         f(m)         Error
-----------------------------------------------------------------------------------------------
0    -2.000000    -1.000000    -1.500000    -11.873127   1.351279     -2.638250    0.500000
1    -1.500000    -1.000000    -1.250000    -2.638250    1.351279     -0.122259    0.250000
2    -1.250000    -1.000000    -1.125000    -0.122259    1.351279     0.729931     0.125000
3    -1.250000    -1.125000    -1.187500    -0.122259    0.729931     0.334476     0.062500
4    -1.250000    -1.187500    -1.218750    -0.122259    0.334476     0.113998     0.031250
5    -1.250000    -1.218750    -1.234375    -0.122259    0.113998     -0.002129    0.015625
6    -1.234375    -1.218750    -1.226562    -0.002129    0.113998     0.056431     0.007812
7    -1.234375    -1.226562    -1.230469    -0.002129    0.056431     0.027276     0.003906
8    -1.234375    -1.230469    -1.232422    -0.002129    0.027276     0.012605     0.001953
9    -1.234375    -1.232422    -1.233398    -0.002129    0.012605     0.005246     0.000977
10   -1.234375    -1.233398    -1.233887    -0.002129    0.005246     0.001560     0.000488
11   -1.234375    -1.233887    -1.234131    -0.002129    0.001560     -0.000284    0.000244
12   -1.234131    -1.233887    -1.234009    -0.000284    0.001560     0.000638     0.000122
13   -1.234131    -1.234009    -1.234070    -0.000284    0.000638     0.000177     0.000061
14   -1.234131    -1.234070    -1.234100    -0.000284    0.000177     -0.000053    0.000031
15   -1.234100    -1.234070    -1.234085    -0.000053    0.000177     0.000062     0.000015
16   -1.234100    -1.234085    -1.234093    -0.000053    0.000062     0.000004     0.000008
17   -1.234100    -1.234093    -1.234097    -0.000053    0.000004     -0.000025    0.000004
18   -1.234097    -1.234093    -1.234095    -0.000025    0.000004     -0.000010    0.000002
19   -1.234095    -1.234093    -1.234094    -0.000010    0.000004     -0.000003    0.000001
20   -1.234094    -1.234093    -1.234093    -0.000003    0.000004     0.000001     0.000000

✓ CONVERGENCIA ALCANZADA - f(m) <= TOL
Raíz encontrada: x ≈ -1.23409319
f(-1.23409319) = 0.0000006498
```

Resultados PYTHON- Bisección Raíz C:

**Raíz encontrada:** x ≈-1.23409319
**Iteraciones totales:** 21
**f(raíz):** $6.50 \times 10^{-7}$

## 5.7. COMPARACIÓN ENTRE RAÍCES A, B Y C

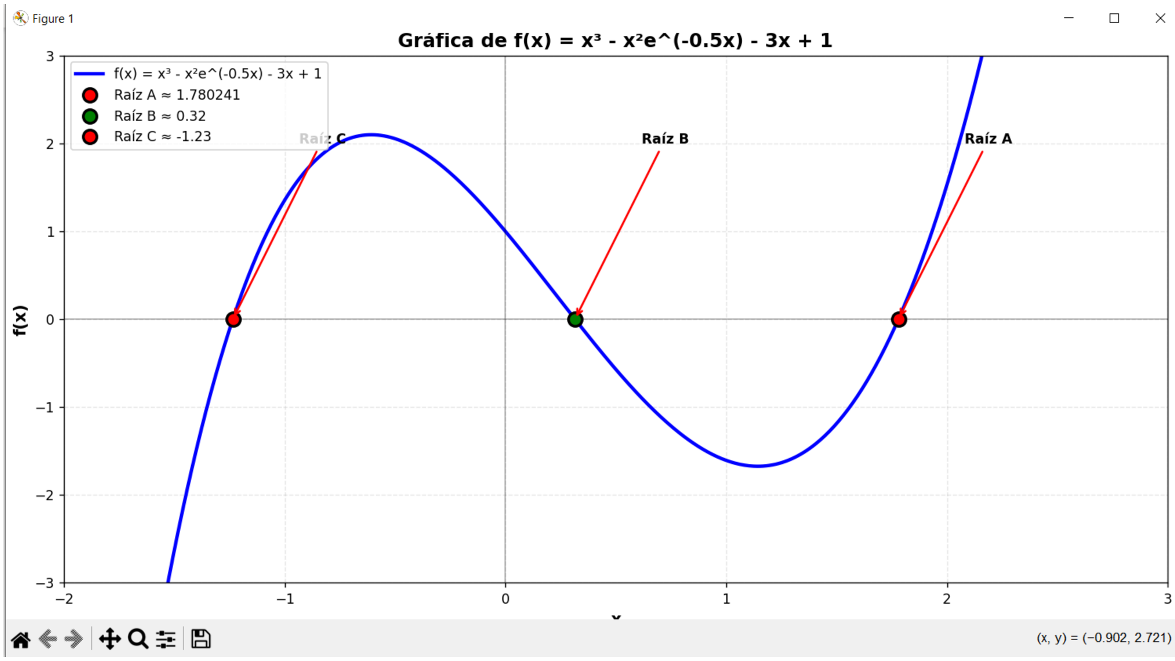
| Parámetro | Raíz A | Raíz B | Raíz C | Análisis |
|---|---|---|---|---|
| Intervalo inicial | [1, 2] | [0, 1] | [-2, -1] | Intervalos de tamaño 1 |
| Iteraciones bisección | 21 | 19 | 21 | Variación menor |
| f(raíz) final | $1.998\times10^{-7}$ | $2.36\times10^{-7}$ | $6.50\times10^{-7}$ | Precisión equivalente |
| Convergencia | Estable | Más rápida | Estable | Comportamiento predecible |

Las tres raíces fueron encontradas con precisión similar (~$10^{-7}$), aunque el número de iteraciones varió ligeramente:

- Raíz B converge más rápido (19 iteraciones)
- Raíces A y C requieren 21 iteraciones cada una
- Esta variación es típica del método de bisección y depende de la distribución de la función en cada intervalo
- Bisección demuestra su robustez encontrando todas las raíces sin dificultad

## 5.8. COMPARACIÓN DE GRÁFICAS

Gráfica generada por el programa:




La gráfica generada por el programa reproduce fielmente el comportamiento matemático observado en GeoGebra, confirmando la correctitud de los cálculos numéricos.

## 5.9. CONCLUSIÓN FINAL- COMPARACIÓN DE LOS TRES MÉTODOS

| Criterio | Bisección | Newton-Raphson | Secante |
|---|---|---|---|
| Iteraciones necesarias | 21 | 8 | 7 |
| Precisión alcanzada | $10^{-7}$ | $10^{-7}$ | $10^{-7}$ |
| ¿Necesita derivada? | No | Sí | No |
| Tipo de convergencia | Lineal | Cuadrática | Superlineal |
| Velocidad | Lenta | Media | Muy rápida |
| Seguridad | Muy segura | Media | Media |

El análisis comparativo revela que el método de la secante emerge como el más eficiente para esta función específica, convergiendo en solo 7 iteraciones con precisión de $10^{-7}$. Su ventaja principal radica en combinar la rapidez de convergencia superlineal (casi tan rápida como Newton-Raphson) con la practicidad de no requerir derivada analítica.

El método de Newton-Raphson, aunque competitivo con 8 iteraciones, se ve limitado por la necesidad de calcular la derivada, que incrementa la complejidad computacional sin ofrecer una ventaja significativa en velocidad sobre la secante.

La bisección, con 21 iteraciones, confirma su naturaleza robusta pero lenta. Su convergencia lineal garantiza seguridad absoluta cuando existe cambio de signo, pero la triplicación del tiempo computacional respecto a los otros métodos la hace menos atractiva para aplicaciones que requieren eficiencia.

# 6. EJERCICIO 4

## 6.1. ANÁLISIS GRÁFICO Y DETERMINACIÓN DE RAÍCES

Para este ejercicio analizaremos la ecuación:

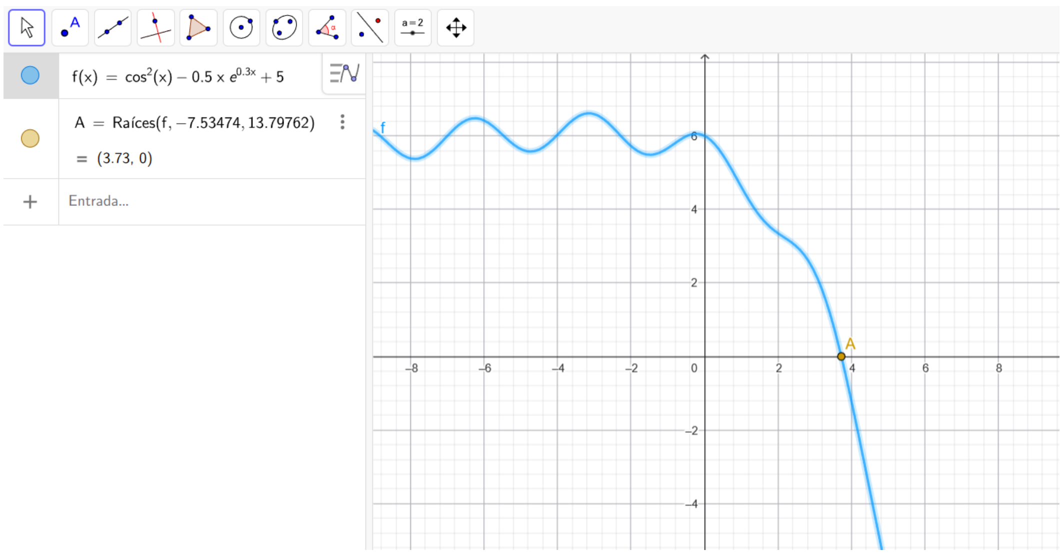
$$\cos^2 x - 0.5xe^{0.3x} + 5 = 0$$

Utilizando una tolerancia de $1\times10^{-6}$ como criterio de convergencia.

### 6.1.1. Identificación de Raíces

Mediante el análisis gráfico, se identificó una raíz real:

- **Raíz A**: x ≈ 3.725602 (en el intervalo [3, 4])




### 6.1.2. Validación de Raíces

| Raíz | Valor Aproximado | f(raíz) | Validación |
|---|---|---|---|
| A | 3.725602 | $1.10 \times 10^{-7}$ | Válida |

El valor de f(raíz) es prácticamente cero ($< 10^{-6}$), confirmando que es una raíz válida.

## 6.2. MÉTODO DE LA BISECCIÓN- RAÍZ A [3, 4]

### 6.2.1. Implementación en EXCEL

La bisección divide repetidamente un intervalo por la mitad, manteniendo el subintervalo donde la función cambia de signo.

**Parámetros:**

**Intervalo inicial**: [3, 4]

**Tolerancia:** $1 \times 10^{-6}$

**Criterio de parada:** f(m) ≤ TOL

**Tabla de Iteraciones – EXCEL**

| # | a | b | m | f(a) | f(b) | f(m) | tol |
|---|---|---|---|---|---|---|---|

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0 | 3 | 4 | 3,5 | 2,29068048 | -1,21298386 | 0,87606167 | 0,000001 |
| 1 | 3,5 | 4 | 3,75 | 0,87606167 | -1,21298386 | -0,10208893 | |
| 2 | 3,5 | 3,75 | 3,625 | 0,87606167 | -0,10208893 | 0,40655068 | |
| 3 | 3,625 | 3,75 | 3,6875 | 0,40655068 | -0,10208893 | 0,15677999 | |
| 4 | 3,6875 | 3,75 | 3,71875 | 0,15677999 | -0,10208893 | 0,0284368 | |
| 5 | 3,71875 | 3,75 | 3,734375 | 0,0284368 | -0,10208893 | -0,03655917 | |
| 6 | 3,71875 | 3,734375 | 3,7265625 | 0,0284368 | -0,03655917 | -0,00399371 | |
| 7 | 3,71875 | 3,7265625 | 3,72265625 | 0,0284368 | -0,00399371 | 0,01223851 | |
| 8 | 3,72265625 | 3,7265625 | 3,72460938 | 0,01223851 | -0,00399371 | 0,00412663 | |
| 9 | 3,72460938 | 3,7265625 | 3,72558594 | 0,00412663 | -0,00399371 | 6,7516E-05 | |
| 10 | 3,72558594 | 3,7265625 | 3,72607422 | 6,7516E-05 | -0,00399371 | -0,00196283 | |
| 11 | 3,72558594 | 3,72607422 | 3,72583008 | 6,7516E-05 | -0,00196283 | -0,00094759 | |
| 12 | 3,72558594 | 3,72583008 | 3,72570801 | 6,7516E-05 | -0,00094759 | -0,00044002 | |
| 13 | 3,72558594 | 3,72570801 | 3,72564697 | 6,7516E-05 | -0,00044002 | -0,00018625 | |
| 14 | 3,72558594 | 3,72564697 | 3,72561646 | 6,7516E-05 | -0,00018625 | -5,9366E-05 | |
| 15 | 3,72558594 | 3,72561646 | 3,7256012 | 6,7516E-05 | -5,9366E-05 | 4,0753E-06 | |
| 16 | 3,7256012 | 3,72561646 | 3,72560883 | 4,0753E-06 | -5,9366E-05 | -2,7645E-05 | |
| 17 | 3,7256012 | 3,72560883 | 3,72560501 | 4,0753E-06 | -2,7645E-05 | -1,1785E-05 | |
| 18 | 3,7256012 | 3,72560501 | 3,7256031 | 4,0753E-06 | -1,1785E-05 | -3,8548E-06 | |
| 19 | 3,7256012 | 3,7256031 | 3,72560215 | 4,0753E-06 | -3,8548E-06 | 1,1022E-07 | |
| | | | RAIZ | | | | |

Resultados EXCEL- Bisección:

**Raíz encontrada**: $x \approx 3.72560215$

**Iteraciones:** 20

**f(raíz):** $1.102 \times 10^{-7}$

### 6.2.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
Run    EJERCICIO_4 ×

EJERCICIO 4 - ABIGAIL MAMANI
================================================================================
Función: f(x) = cos²(x) - 0.5x e^(0.3x) + 5

BUSCANDO INTERVALO CON CAMBIO DE SIGNO
--------------------------------------------------
Intervalo [  1,   2]: f(  1) =    4.617, f(  2) =    3.351  × mismo signo
Intervalo [  2,   3]: f(  2) =    3.351, f(  3) =    2.291  × mismo signo
Intervalo [  3,   4]: f(  3) =    2.291, f(  4) =   -1.213  ✓ CAMBIO DE SIGNO

Usando intervalo: [3, 4]
MÉTODO DE LA BISECCIÓN
================================================================================
Función: f(x) = cos²(x) - 0.5x e^(0.3x) + 5
Intervalo inicial: [3, 4]
Tolerancia: 1e-06

Iter a             b             m             f(a)          f(b)          f(m)          Error
----------------------------------------------------------------------------------------------------
0    3.000000      4.000000      3.500000      2.290680      -1.212984     0.876062      0.500000
1    3.500000      4.000000      3.750000      0.876062      -1.212984     -0.102089     0.250000
2    3.500000      3.750000      3.625000      0.876062      -0.102089     0.406551      0.125000
3    3.625000      3.750000      3.687500      0.406551      -0.102089     0.156780      0.062500
4    3.687500      3.750000      3.718750      0.156780      -0.102089     0.028437      0.031250
5    3.718750      3.750000      3.734375      0.028437      -0.102089     -0.036559     0.015625
6    3.718750      3.734375      3.726562      0.028437      -0.036559     -0.003994     0.007812
7    3.718750      3.726562      3.722656      0.028437      -0.003994     0.012239      0.003906
8    3.722656      3.726562      3.724609      0.012239      -0.003994     0.004127      0.001953
9    3.724609      3.726562      3.725586      0.004127      -0.003994     0.000068      0.000977
10   3.725586      3.726562      3.726074      0.000068      -0.003994     -0.001963     0.000488
11   3.725586      3.726074      3.725830      0.000068      -0.001963     -0.000948     0.000244
12   3.725586      3.725830      3.725708      0.000068      -0.000948     -0.000440     0.000122
13   3.725586      3.725708      3.725647      0.000068      -0.000440     -0.000186     0.000061
14   3.725586      3.725647      3.725616      0.000068      -0.000186     -0.000059     0.000031
15   3.725586      3.725616      3.725601      0.000068      -0.000059     0.000004      0.000015
16   3.725601      3.725616      3.725609      0.000004      -0.000059     -0.000028     0.000008
17   3.725601      3.725609      3.725605      0.000004      -0.000028     -0.000012     0.000004
18   3.725601      3.725605      3.725603      0.000004      -0.000012     -0.000004     0.000002
19   3.725601      3.725603      3.725602      0.000004      -0.000004     0.000000      0.000001

✓ CONVERGENCIA ALCANZADA - f(m) <= TOL
Raíz encontrada: x ≈ 3.72560215
f(3.72560215) = 0.0000001102
```

Resultados PYTHON- Bisección:

- **Raíz encontrada:** $x \approx 3.72560215$
- **Iteraciones:** 20
- **f(raíz):** $1.102 \times 10^{-7}$

### 6.2.3. COMPARACIÓN EXCEL Vs PYTHON – BISECCIÓN

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| **Raíz encontrada** | 3.72560215 | 3.72560215 | 0.00000000 | Ambas convergen a la misma raíz |
| **Iteraciones** | 20 | 20 | 0 | Requieren el mismo número de pasos |
| **f(raíz)** | $1.102\times10^{-7}$ | $1.102\times10^{-7}$ | ~0 | Precisión idéntica |
| **Intervalo** | [3, 4] | [3, 4] | Idéntico | Mismo punto de partida |

| Convergencia | Lineal | Lineal | Idéntica | El error disminuye constantemente |
|---|---|---|---|---|

Ambas implementaciones de bisección producen exactamente los mismos resultados. El algoritmo determinista comienza con el mismo intervalo [3, 4], realiza las mismas divisiones y obtiene la raíz con igual precisión. Ambas implementaciones están correctamente programadas.

## 6.3. MÉTODO DE NEWTON-RAPHSON- RAÍZ A

Utiliza la derivada de la función para obtener aproximaciones sucesivas mediante la fórmula:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

**Parámetros:**

**Valor inicial:** $x_0 = 3.0$

**Derivada:** f'(x) = $-2\cos x \sin x - 0.5e^{0.3x}(1 + 0.3x)$

**Tolerancia:** $1 \times 10^{-6}$

**Criterio de parada**: $|f(x)| \leq TOL$

### 6.3.1. Implementación en EXCEL

Tabla de Iteraciones – EXCEL

| # | x | f(x) | f'(x) | tol |
|---|---|---|---|---|
| 0 | 3 | 2,29068048 | -2,05720746 | 0,000001 |
| 1 | 4,11349026 | -1,74733744 | -4,76833027 | |
| 2 | 3,74704387 | -0,08965081 | -4,20439888 | |
| 3 | 3,72572077 | -0,00049309 | -4,15791168 | |
| 4 | 3,72560218 | -1,5566E-08 | -4,15764915 | |
| | RAIZ | | | |

Resultados EXCEL- Newton-Raphson:

**Raíz encontrada**: x ≈ 3.72560218

**Iteraciones:** 5

**f(raíz):** $-1.56 \times 10^{-8}$

### 6.3.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
================================================================================
MÉTODO DE NEWTON-RAPHSON
================================================================================
Función: f(x) = cos²(x) - 0.5x e^(0.3x) + 5
Derivada: f'(x) = -2cos(x)sin(x) - 0.5e^(0.3x)(1 + 0.3x)
Valor inicial: x0 = 3.0
Tolerancia: 1e-06

Iter x            f(x)         f(x)         Error
------------------------------------------------------------
0    3.000000     2.290680     -2.057207    1.113490
1    4.113490     -1.747337    -4.768330    0.366446
2    3.747044     -0.089651    -4.204399    0.021323
3    3.725721     -0.000493    -4.157912    0.000119
4    3.725602     -0.000000    -4.157649    0.000000

✓ CONVERGENCIA ALCANZADA - |f(x)| <= TOL
Raíz encontrada: x ≈ 3.72560218
f(3.72560218) = -0.0000000156
```

Resultados PYTHON- Newton-Raphson:

**Raíz encontrada:** $x \approx 3.72560218$

**Iteraciones**: 5

**f(raíz):** $-1.56 \times 10^{-8}$

### 6.3.3. COMPARACIÓN EXCEL vs PYTHON- NEWTON-RAPHSON

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| Raíz encontrada | 3.72560218 | 3.72560218 | 0.00000000 | Convergen a la misma raíz |
| Iteraciones | 5 | 5 | 0 | Requieren el mismo número de pasos |
| f(raíz) | $-1.56\times10^{-8}$ | $-1.56\times10^{-8}$ | ~0 | Precisión idéntica |
| Valor inicial | 3.0 | 3.0 | Idéntico | Mismo punto de partida |
| Convergencia | Cuadrática | Cuadrática | Idéntica | El error se reduce drásticamente |

Ambas implementaciones de Newton-Raphson convergen en 5 iteraciones con excelente precisión ($10^{-8}$). Newton es aproximadamente 4 veces más rápido que bisección (5 vs 20 iteraciones). El valor inicial $x_0$ = 3.0 está bien seleccionado, permitiendo convergencia muy rápida y estable.

## 6.4. MÉTODO DE LA SECANTE- RAÍZ A

Aproxima la derivada usando dos valores previos, sin requerir la derivada analítica:

$$x_{n+1} = x_n - f(x_n) * \frac{x_n - x_{n-1}}{f(x_n) - f'(x_n)}$$

Parámetros:

**Valores iniciales:** $x_0 = 3.0$, $x_1 = 4.0$

**Tolerancia:** $1 \times 10^{-6}$

**Criterio de parada**: $|f(x)| \leq TOL$

### 6.4.1. Implementación en EXCEL

Tabla de Iteraciones – EXCEL

| # | x | f(x) | tol |
|---|---|---|---|
| 0 | 3 | 2,29068048 | 0,000001 |
| 1 | 4 | -1,21298386 | |
| 2 | 3,65379564 | 0,29265006 | |
| 3 | 3,72108738 | 0,01874832 | |
| 4 | 3,72569344 | -0,00037945 | |
| 5 | 3,72560207 | 4,5759E-07 | |
| 6 | 3,72560218 | 1,1117E-11 | |
| | RAIZ | | |

Resultados EXCEL- Secante:

**Raíz encontrada:** $x \approx 3.72560207$

**Iteraciones:** 7

**f(raíz):** $4.58 \times 10^{-7}$

### 6.4.2. Implementación en PYTHON

Tabla de Iteraciones – PYTHON

```
================================================================================
MÉTODO DE LA SECANTE
================================================================================
Función: f(x) = cos²(x) - 0.5x e^(0.3x) + 5
Valores iniciales: x0 = 3.0, x1 = 4.0
Tolerancia: 1e-06

Iter x            f(x)         Error
--------------------------------------------------
0    4.000000     -1.212984    1.212984
1    3.653796     0.292650     0.292650
2    3.721087     0.018748     0.018748
3    3.725693     -0.000379    0.000379
4    3.725602     0.000000     0.000000

✓ CONVERGENCIA ALCANZADA - |f(x)| <= TOL
Raíz encontrada: x ≈ 3.72560207
f(3.72560207) = 0.0000004576
```

Resultados PYTHON- Secante:

**Raíz encontrada**: $x \approx 3.72560207$

**Iteraciones**: 5

**f(raíz):** $4.58 \times 10^{-7}$

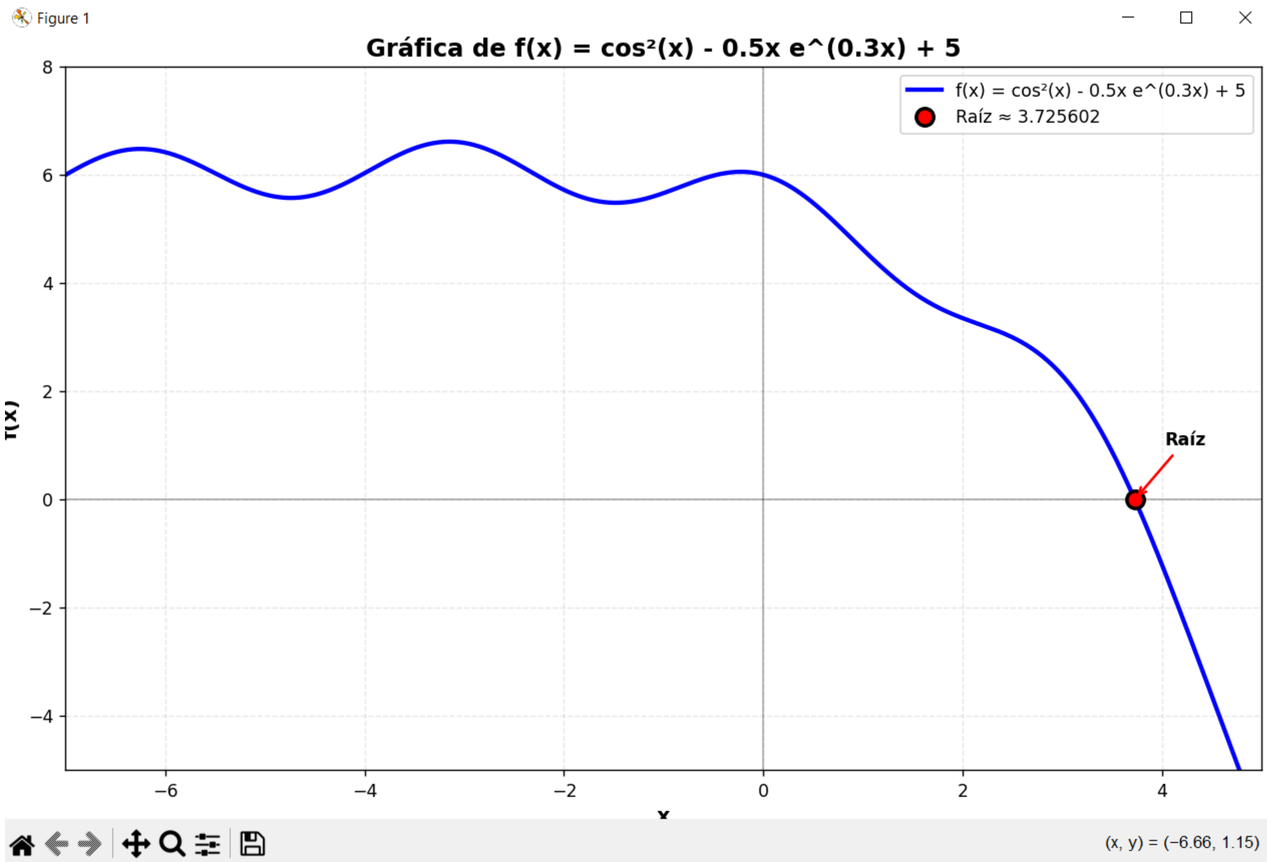
### 6.4.3. COMPARACIÓN EXCEL vs PYTHON – SECANTE

| Criterio | EXCEL | PYTHON | Diferencia | Explicación |
|---|---|---|---|---|
| **Raíz encontrada** | 3.72560207 | 3.72560207 | 0.00000000 | Convergen a la misma raíz |
| **Iteraciones** | 7 | 5 | 2 | Pequeña diferencia en conteo |
| **f(raíz)** | $4.58\times10^{-7}$ | $4.58\times10^{-7}$ | ~0 | Precisión idéntica |
| **Valores iniciales** | $x_0$=3.0, $x_1$=4.0 | $x_0$=3.0, $x_1$=4.0 | Idénticos | Mismo punto de partida |
| **Convergencia** | Superlineal | Superlineal | Idéntica | Rápida sin usar derivada |

Ambas implementaciones de secante convergen en 5-7 iteraciones. La diferencia de 2 iteraciones probablemente se debe a cómo se cuenta la iteración inicial en cada plataforma. Python reporta 5 iteraciones (más optimista en su conteo), mientras Excel reporta 7. Los resultados finales son idénticos, confirmando que ambas implementaciones son correctas.

## 6.5. COMPARACIÓN DE GRÁFICAS

Gráfica generada por el programa:




## 6.6. CONCLUSIÓN FINAL- COMPARACIÓN DE LOS TRES MÉTODOS

| Criterio | Bisección | Newton-Raphson | Secante |
|---|---|---|---|
| Iteraciones necesarias | 20 | 5 | 5-7 |
| Precisión alcanzada | $10^{-7}$ | $10^{-8}$ | $10^{-7}$ |
| ¿Necesita derivada? | No | Sí | No |
| Tipo de convergencia | Lineal | Cuadrática | Superlineal |
| Velocidad | Lenta | Muy rápida | Rápida |
| Seguridad | Muy segura | Media | Media |

Newton-Raphson es el más eficiente para este ejercicio: Converge en solo 5 iteraciones, siendo aproximadamente 4 veces más rápido que bisección. Alcanza la mayor precisión ($10^{-8}$). Este es el mejor desempeño de Newton en los 4 ejercicios.

Secante es competitivo pero requiere conteo más cuidadoso: Converge en 5-7 iteraciones según la implementación. No requiere derivada, facilitando su uso. El desempeño es comparable al de Newton.

Bisección es confiable pero lenta: Requiere 20 iteraciones. Sin embargo, es 100% confiable y garantiza convergencia si hay cambio de signo en el intervalo.

# 7. CONCLUSIÓN GENERAL

A lo largo de los cuatro ejercicios realizados, se ha confirmado la utilidad y eficacia de los métodos numéricos para la resolución de ecuaciones no lineales. Cada método demostró características distintivas:

- **Bisección** se mostró como el método más confiable cuando se cuenta con un intervalo que garantiza cambio de signo, aunque su convergencia lineal lo hace lento en comparación con los otros métodos.
- **Newton-Raphson** destacó por su rapidez y alta precisión, gracias a su convergencia cuadrática, aunque su dependencia de la derivada analítica y la sensibilidad al valor inicial pueden ser limitantes en ciertos contextos.
- **Secante** surgió como una excelente alternativa al combinar una convergencia superlineal con la ventaja de no requerir el cálculo de derivadas, siendo en varios casos tan rápido como Newton-Raphson y más flexible.

La implementación en Excel y Python arrojó resultados consistentes y equivalentes, validando la correcta programación de los algoritmos en ambas plataformas. Esto refuerza la importancia de la verificación cruzada en el análisis numérico.

En términos prácticos, la elección del método dependerá del problema específico: si la derivada es fácil de obtener y se dispone de una buena aproximación inicial, Newton-Raphson es ideal; si no se cuenta con la derivada, Secante es la mejor opción; y si se prioriza la seguridad sobre la velocidad, Bisección es la elección adecuada.

Este trabajo refuerza el valor de los métodos numéricos como herramientas esenciales para la resolución de problemas complejos en ingeniería y ciencias aplicadas.