Actividad 4: Resolución de Sistemas de Ecuaciones No Lineales.

1. Ejercicios y Desarrollo

Se plantearon 10 sistemas de ecuaciones no lineales, 5 con dos variables y 5 con tres variables. Para cada sistema definir las funciones correspondientes en *Python*, utilizar la función *fsolve* de la librería *scipy*. En el caso de los sistemas con dos variables realizar gráficas de las curvas que representan las ecuaciones, de manera que las soluciones aparezcan como puntos de intersección.

2. Sistemas de Ecuaciones No lineales de 2 variables

1. $$x^2 + y^2 = 4, \quad e^x + y = 1$$

2. $$\sin(x) + y^2 = 1, \quad x^2 + y = 2$$

3. $$x^3 - y = 0, \quad x + y^2 = 4$$

4. $$e^x + y = 3, \quad x^2 + y^2 = 5$$

5. $$\ln(x+2) + y = 1, \quad x^2 - y = 2$$

a. Realizar Pseudocódigo

4-

INICIO

Definir sistema de ecuaciones ej(x, y):
    ecuacion1 = e^x + y - 3
    ecuacion2 = x^2 + y^2 - 5
    RETORNAR [ecuacion1, ecuacion2]

Definir punto inicial = (1, 1)
Aplicar método numérico (ej. fsolve) con función ej y punto inicial
Guardar la solución en sol

Mostrar en pantalla "Ejercicio 4:", sol

--- Preparar malla de puntos para graficar ---
Definir rango de x entre -3 y 3
Definir rango de y entre -3 y 3
Construir malla (X, Y) con estos rangos

```
# --- Graficar curvas ---
Dibujar la curva e^X + Y - 3 = 0 en color rojo
Dibujar la curva X^2 + Y^2 - 5 = 0 en color azul

# --- Marcar solución ---
Dibujar punto (sol.x, sol.y) en color verde

# --- Ajustes del gráfico ---
Colocar título "Ejercicio 4"
Etiquetar ejes X e Y
Activar cuadrícula
Mostrar la gráfica en pantalla
```

FIN

5-

```
INICIO

Definir sistema de ecuaciones ej(x, y):
   ecuacion1 = log(x + 2) + y - 1
   ecuacion2 = x^2 - y - 2
   RETORNAR [ecuacion1, ecuacion2]

# --- Resolver sistema ---
Definir punto inicial = (1, 1)
Aplicar método numérico (ej. fsolve) con función ej y punto inicial
Guardar la solución en sol

Mostrar en pantalla "Ejercicio 5:", sol

# --- Preparar espacio de puntos para graficar ---
Definir rango de x desde -3 hasta 3
Definir rango de y desde -3 hasta 3
Construir malla (X, Y) con esos rangos

# --- Graficar ecuaciones ---
Graficar la curva log(X+2) + Y - 1 = 0 en color rojo
Graficar la curva X^2 - Y - 2 = 0 en color azul

# --- Marcar solución ---
Dibujar punto (sol.x, sol.y) en color verde

# --- Ajustes del gráfico ---
Colocar título "Ejercicio 5"
Etiquetar ejes X e Y
Activar cuadrícula
Mostrar la gráfica en pantalla
```

FIN

b. Codificar en Python

4-

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import fsolve
def ej(vars):
   x, y = vars
   return [ np.exp(x) + y - 3, x**2 + y**2 -5]
plt.show()

# Resolver
sol = fsolve(ej, [1, 1])
print("Ejercicio 4:", sol)

#graficar
x = np.linspace(-3, 3, 400)
y = np.linspace(-3, 3, 400)
X, Y = np.meshgrid(x, y)

plt.contour(X, Y, np.exp(X) + Y - 3, [0], colors='r')
plt.contour(X, Y, X**2 + Y**2 - 5,[0], colors='b')
plt.plot(sol[0], sol[1], 'go')
plt.title("Ejercicio 4")
plt.xlabel("x");
plt.ylabel("y")
plt.grid(True);
plt.show()
```

5-

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import fsolve

def ej(vars):
   x, y = vars
   return [np.log(x+2)+y-1, x**2 - y - 2]
plt.show()
# Resolver
sol = fsolve(ej, [1, 1])
print("Ejercicio 5:", sol)

#graficar
x = np.linspace(-3, 3, 400)
y = np.linspace(-3, 3, 400)
```

```
X, Y = np.meshgrid(x, y)

plt.contour(X, Y, np.log(X+2)+Y-1, [0], colors='r')
plt.contour(X, Y, X**2 - Y - 2, [0], colors='b')
plt.plot(sol[0], sol[1], 'go')
plt.title("Ejercicio 5")
plt.xlabel("x");
plt.ylabel("y")
plt.grid(True);
plt.show()
```

3. Sistemas de Ecuaciones No lineales de 3 variables

6. $x^2 + y + z = 4, \quad x + y^2 + z = 5, \quad x + y + z^2 = 6$

7. $e^x + y + z = 7, \quad x^2 + y^2 = 4, \quad z - y = 1$

8. $x^2 + y^2 + z^2 = 9, \quad xy = 2, \quad x + z = 3$

9. $\sin(x) + y = 2, \quad y^2 + z = 3, \quad x + z^2 = 4$

10. $x^2 + y = 1, \quad y^2 + z = 2, \quad z^2 + x = 3$

c. Realizar Pseudocódigo

9-

```
INICIO
Definir sistema de ecuaciones ej1(x, y, z):
   ecuacion1 = sen(x) + y - 2
   ecuacion2 = y^2 + z - 3
   ecuacion3 = x + z^2 - 4
   RETORNAR [ecuacion1, ecuacion2, ecuacion3]
# --- Resolver sistema ---
Definir punto inicial = (1, 1, 1)
Aplicar método numérico (ej. fsolve) con función ej1 y punto inicial
Guardar la solución en sol
Mostrar en pantalla:
   "Ejercicio 9 - solución aproximada (x, y, z) =", sol
FIN
```

10-

```
INICIO

Definir sistema de ecuaciones ej1(x, y, z):
   ecuacion1 = x^2 + y - 1
   ecuacion2 = y^2 + z - 2
   ecuacion3 = z^2 + x - 3
   RETORNAR [ecuacion1, ecuacion2, ecuacion3]

# --- Resolver sistema ---
Definir punto inicial = (1, 1, 1)
Aplicar método numérico (ej. fsolve) con función ej1 y punto inicial
Guardar la solución en sol

Mostrar en pantalla:
   "Ejercicio 9 - solución aproximada (x, y, z) =", sol

FIN
```

d. Codificar en Python (ejercicios 9 y 10 usar función DEF)

9-

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import fsolve

def ej1(vars):
   x, y,z = vars
   return [np.sin(x) + y - 2, y**2 + z - 3, x + z**2 - 4]

sol=fsolve(ej1, [1, 1, 1])
print("Ejercicio 9 - solucion aproximada (x,y,z) =", sol)
```

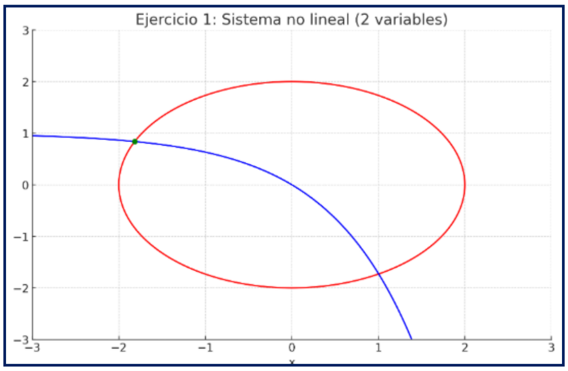
10-

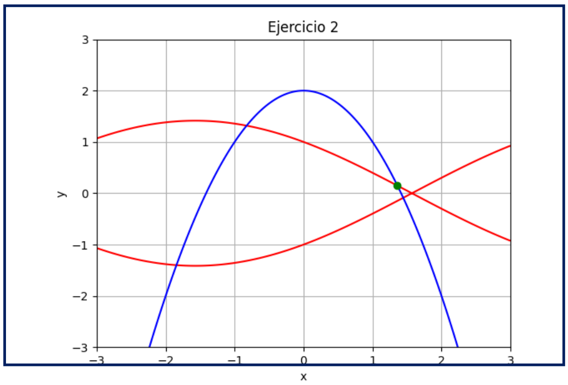
```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import fsolve

def ej1(vars):
   x, y,z = vars
   return [x**2 + y - 1, y**2 + z - 2, z**2 + x - 3]

sol=fsolve(ej1, [1, 1, 1])
print("Ejercicio 9 - solucion aproximada (x,y,z) =", sol)
```

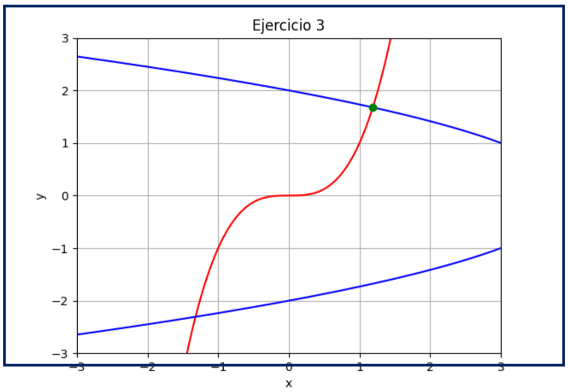
4. Resultados (Máximo 10 líneas en texto)

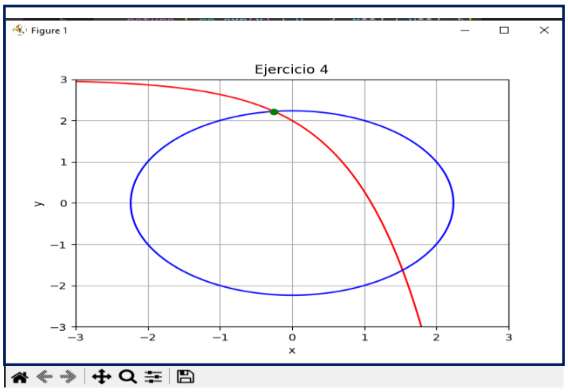
Ejercicio 1: [-1.81626407 0.8373678]
Ejercicio 2: [1.36080308 0.14821497]
Ejercicio 3: [1.18804969 1.67688709]
Ejercicio 4: [-0.25095313 2.22194116]
Ejercicio 5: [1.33947269 -0.20581292]
Ejercicio 6 - solucion aproximada (x,y,z) = [0.73442414 1.52710987 1.93351131]
Ejercicio 7 - solucion aproximada (x,y,z) = [1.29891043 1.23709145 1.80430383]
Ejercicio 8- solucion aproximada (x,y,z) = [1.04053051 1.4437803 0.83279472]
Ejercicio 9 - solucion aproximada (x,y,z) = [1.05020209 1.13247624 1.71749757]
Ejercicio 10 - solucion aproximada (x,y,z) = [0.57785046 0.66608884 1.55632565]

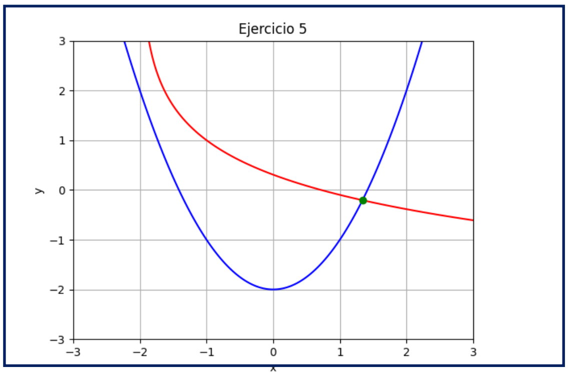
5. Pegar las 5 gráficas generadas de las ecuaciones No lineales de 2 variables (Que encajen en los recuadros.











6. Conclusiones (Máximo en 10 líneas)

El uso de herramientas computacionales como Python, junto con librerías especializadas (NumPy, Matplotlib y SciPy), permitió resolver de manera eficiente distintos sistemas de ecuaciones no lineales de dos y tres variables. Mediante la función fsolve se obtuvieron soluciones aproximadas que, en muchos casos, serían difíciles o imposibles de hallar de forma analítica.

La representación gráfica de los sistemas con curvas y superficies facilitó la interpretación visual de los resultados, mostrando cómo se determinan los puntos de intersección entre las ecuaciones planteadas. Esto evidencia la importancia de combinar métodos numéricos con recursos gráficos para el análisis de problemas matemáticos complejos.