

Escuela Politécnica Nacional

Nombre: Francisco Ulloa

Fecha: Quito, 5 de enero de 2026

Tema: Eliminación Gaussiana vs Gauss-Jordan

Repositorio:

https://github.com/Fu5CHAR/Metodos_numericos_2025B_Ulloa-Francisco/tree/main

In [2]: `from tarea09_funciones import *`

```
[01-07 17:25:57][INFO] 2026-01-07 17:25:57.281348
[01-07 17:25:57][INFO] 2026-01-07 17:25:57.282270
[01-07 17:25:57][INFO] 2026-01-07 17:25:57.283376
[01-07 17:25:57][INFO] 2026-01-07 17:25:57.285525
[01-07 17:25:57][INFO] 2026-01-07 17:25:57.282270
[01-07 17:25:57][INFO] 2026-01-07 17:25:57.283376
[01-07 17:25:57][INFO] 2026-01-07 17:25:57.285525
```

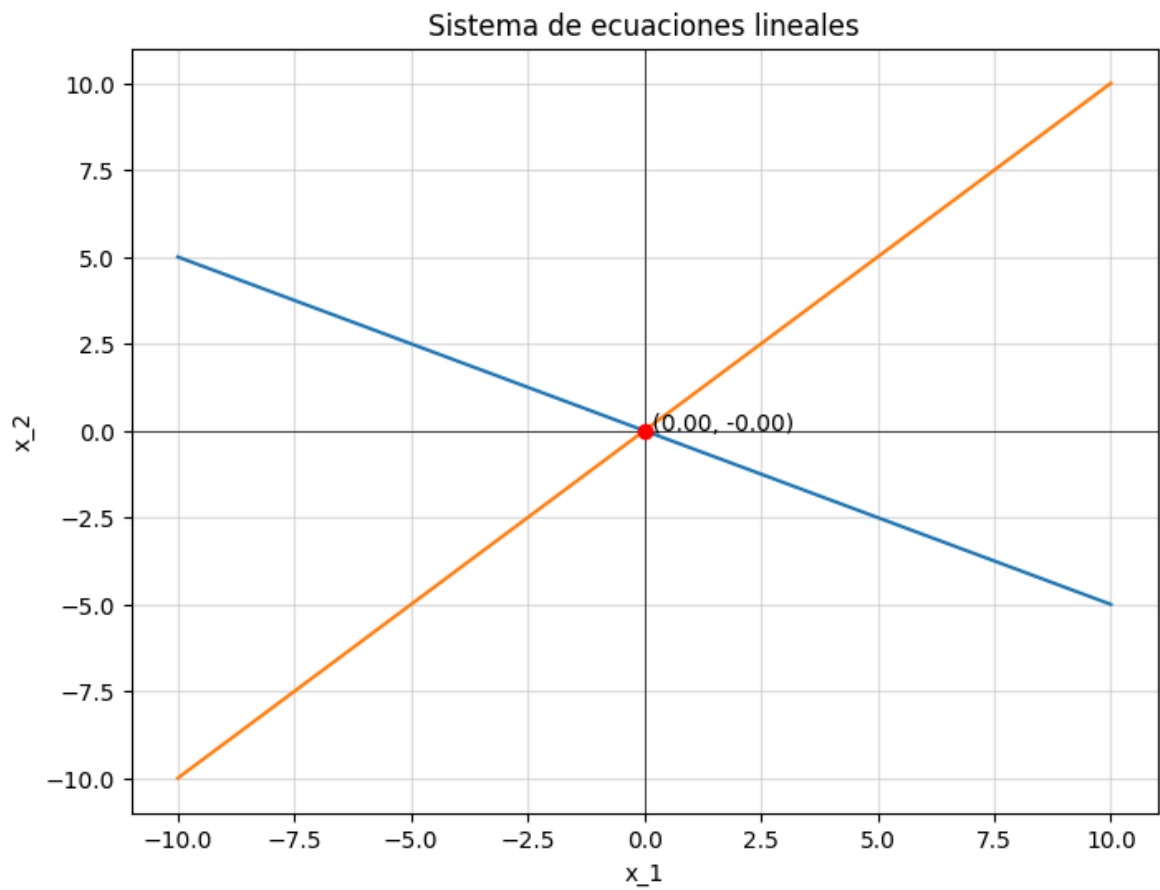
1. Para cada uno de los siguientes sistemas, lineales, obtenga, de ser posible, una solución con métodos gráficos. Explique los resultados desde un punto de vista geométrico.

a.

$$x_1 + x_2 = 0$$

$$x_1 - x_2 = 0$$

In [21]: `resolver_y_graficar("x_1 + 2*x_2 = 0", "x_1 - x_2 = 0")`

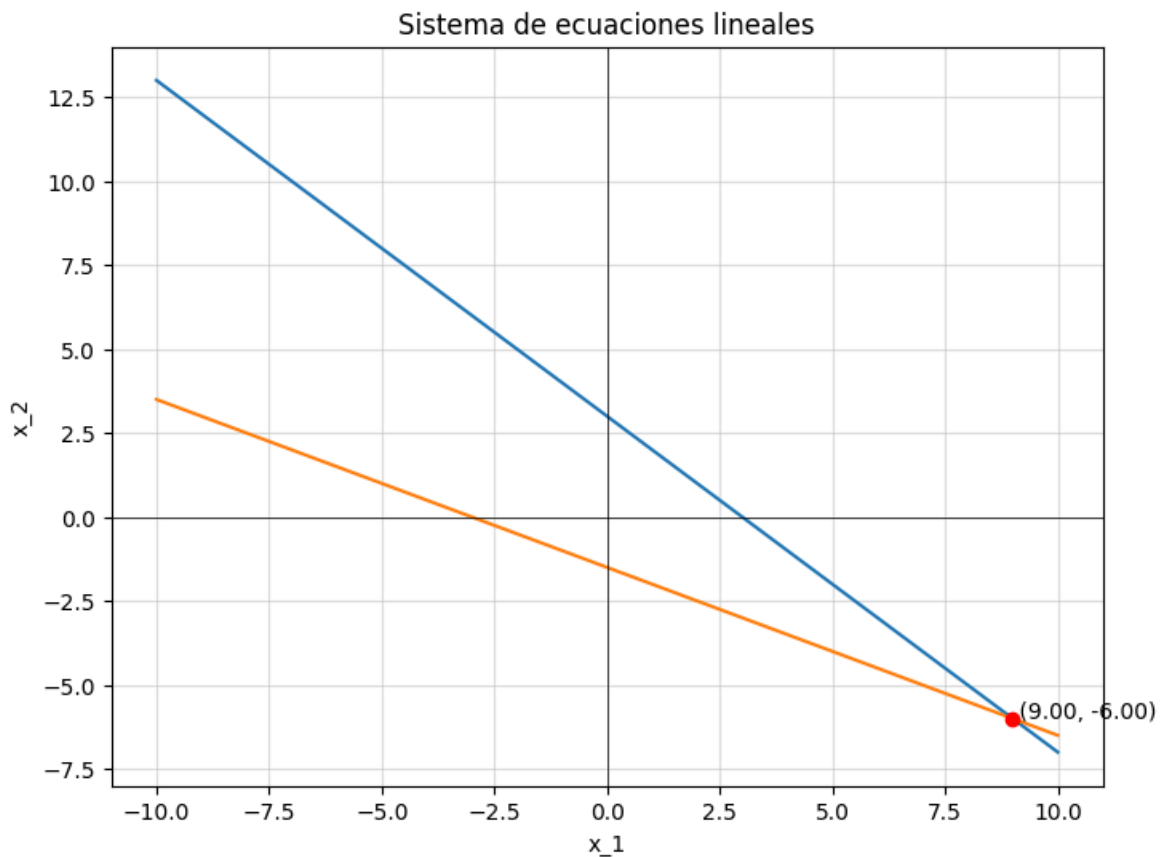


b.

$$x_1 + x_2 = 3$$

$$-2x_1 - 4x_2 = 6$$

In [22]: `resolver_y_graficar('x_1 + x_2 = 3', '-2*x_1 -4*x_2 = 6')`



c.

$$2x_1 + x_2 = -1$$

$$x_1 + x_2 = 2 \quad x_1 - 3x_2 = 5$$

In [23]: `resolver_y_graficar('2*x_1 + x_2 = -1', 'x_1 + x_2 = 2', 'x_1 - 3*x_2 = 5')`

El sistema no tiene solución única.

d.

$$2x_1 + x_2 + x_3 = 1$$

$$2x_1 + 4x_2 - x_3 = -1$$

In [24]: `resolver_y_graficar('2*x_1 + x_2 + x_3 = 1', '2*x_1 + 4*x_2 - x_3 = -1')`

El sistema no tiene solución única.

Explicación de los resultados obtenidos: Como se pudo ver en los literales a) y b) si obtuvimos una respuesta y por consiguiente una gráfica de dicha resolución dado que, gráficamente las líneas formadas por sus respectivas ecuaciones lineales se intersecan entre si en un solo punto. Para el literal c) y d) tenemos cantides distintas entre ecuaciones y variables, lo cual nos deja con múltiples soluciones y ninguna, respectivamente.

2. Utilice la eliminación gaussiana con sustitución hacia atrás y aritmética de redonde de dos dígitos para resolver los siguientes sistemas lineales. No reordene las ecuaciones

La solución exacta para cada sistema es: $x_1 = -1, x_2 = 2, x_3 = 3$

a.

$$-x_1 + 4x_2 + x_3 = 8$$

$$\frac{5}{3}x_1 + \frac{2}{3}x_2 + \frac{2}{3}x_3 = 11$$

```
In [25]: Ab= [[-1, 4, 1, 8],[5/3, 2/3, 2/3, 1],[2, 1, 4, 11]]
          eliminacion_gaussiana_redondeo(Ab)
```

```
[01-07 17:23:50][INFO]
[[-1.    4.    1.    8. ]
 [-0.    7.35  2.34 14.36]
 [ 0.    9.    6.   27. ]]
[01-07 17:23:50][INFO]
[[-1.    4.    1.    8. ]
 [-0.    7.35  2.34 14.36]
 [ 0.    0.03  3.15  9.48]]
```

```
Out[25]: array([-0.99,  1.   ,  3.01])
```

```
In [26]: b = [-1, 2, 3]
          calcular_error(eliminacion_gaussiana_redondeo(Ab),b)
```

```
[01-07 17:23:50][INFO]
[[-1.    4.    1.    8. ]
 [-0.    7.35  2.34 14.36]
 [ 0.    9.    6.   27. ]]
[01-07 17:23:50][INFO]
[[-1.    4.    1.    8. ]
 [-0.    7.35  2.34 14.36]
 [ 0.    0.03  3.15  9.48]]
```

Error calculado para la variable x 1 : 1.010101010101011 %

Error calculado para la variable x 2 : 100.0 %

Error calculado para la variable x 3 : 0.3322259136212554 %

b.

$$4x_1 + 2x_2 - x_3 = -5$$

$$\frac{1}{9}x_1 + \frac{1}{9}x_2 - \frac{1}{3}x_3 = -1$$

```
In [27]: Cd=[[4, 2, -1, -5],[1/9, 1/9, -1/3, -1],[1, 4, 2, 9]]
          eliminacion_gaussiana_redondeo(Cd)
```

```
[01-07 17:23:50][INFO]
[[ 4.000e+00  2.000e+00 -1.000e+00 -5.000e+00]
 [-1.000e-02  5.000e-02 -3.000e-01 -8.500e-01]
 [ 0.000e+00  3.500e+00  2.250e+00  1.025e+01]]
[01-07 17:23:50][INFO]
[[ 4.000e+00  2.000e+00 -1.000e+00 -5.000e+00]
 [-1.000e-02  5.000e-02 -3.000e-01 -8.500e-01]
 [ 0.000e+00  0.000e+00  2.325e+01  6.975e+01]]
```

```
Out[27]: array([-1.,  1.,  3.])
```

```
In [28]: calcular_error(eliminacion_gaussiana_redondeo(Cd),b)
```

```
[01-07 17:23:50][INFO]
[[ 4.000e+00  2.000e+00 -1.000e+00 -5.000e+00]
 [-1.000e-02  5.000e-02 -3.000e-01 -8.500e-01]
 [ 0.000e+00  3.500e+00  2.250e+00  1.025e+01]]
[01-07 17:23:50][INFO]
[[ 4.000e+00  2.000e+00 -1.000e+00 -5.000e+00]
 [-1.000e-02  5.000e-02 -3.000e-01 -8.500e-01]
 [ 0.000e+00  0.000e+00  2.325e+01  6.975e+01]]
```

Error calculado para la variable x 1 : 0.0 %

Error calculado para la variable x 2 : 100.0 %

Error calculado para la variable x 3 : 0.0 %

3.Utilice el algoritmo de eliminación gaussiana para resolver, de ser posible, los siguientes sistemas lineales, y determine si se necesitan intercambios de fila:

a.

$$x_1 - x_2 + 3x_3 = 2$$

$$3x_1 - 3x_2 + x_3 = -1 \quad x_1 + x_2 = 3$$

```
In [29]: Ba=[[1, -1, 3, 2],[3, -3, 1, -1],[1, 1, 0, 3]]
          eliminacion_gaussiana(Ba)
```

```
[01-07 17:23:50][INFO]
[[ 1. -1.  3.  2.]
 [ 0.  0. -8. -7.]
 [ 0.  2. -3.  1.]]
[01-07 17:23:50][INFO]
[[ 1. -1.  3.  2.]
 [ 0.  2. -3.  1.]
 [ 0.  0. -8. -7.]]
```

Gauss:

Sí se necesitan intercambios de fila
la solución es:

```
Out[29]: array([1.1875, 1.8125, 0.875 ])
```

b.

$$2x_1 - 1.5x_2 + 3x_3 = 1$$

$$-x_1 + 0 + 2x_3 = 3 \quad 4x_1 - 4.5x_2 + 5x_3 = 1$$

```
In [30]: Dc=[[2, -1.5, 3, 1],[-1, 0, 2, 3],[4, -4.5, 5, 1]]
          eliminacion_gaussiana(Dc)
```

```
[01-07 17:23:50][INFO]
[[-1.  0.  2.  3. ]
 [ 0. -1.5 7.  7. ]
 [ 0. -4.5 13. 13. ]]
[01-07 17:23:50][INFO]
[[-1.  0.  2.  3. ]
 [ 0. -1.5 7.  7. ]
 [ 0.  0. -8. -8. ]]
```

Gauss:

Sí se necesitan intercambios de fila
la solución es:

```
Out[30]: array([-1., -0.,  1.])
```

c.

$$2x_1 + 0 + 0 + 0 = 3$$

$$x_1 + 1.5x_2 + 0 + 0 = 4.5 \quad 0 - 3x_2 + 0.5x_3 + 0 = -6.6 \quad 2x_1 - 2x_2 + x_3 + x_4 = 0.8$$

```
In [31]: Ef=[[2,0,0,0,3],[1,1.5,0,0,4.5],[0,-3,0.5,0,-6.6],[2,-2,1,1,0.8]]
          eliminacion_gaussiana(Ef)
```

```
[01-07 17:23:50][INFO]
[[ 1.  1.5  0.  0.  4.5]
 [ 0. -3.  0.  0. -6. ]
 [ 0. -3.  0.5 0. -6.6]
 [ 0. -5.  1.  1. -8.2]]
[01-07 17:23:50][INFO]
[[ 1.  1.5  0.  0.  4.5]
 [ 0. -3.  0.  0. -6. ]
 [ 0.  0.  0.5 0. -0.6]
 [ 0.  0.  1.  1.  1.8]]
[01-07 17:23:50][INFO]
[[ 1.  1.5  0.  0.  4.5]
 [ 0. -3.  0.  0. -6. ]
 [ 0.  0.  0.5 0. -0.6]
 [ 0.  0.  0.  1.  3. ]]
```

Gauss:

Sí se necesitan intercambios de fila
la solución es:

```
Out[31]: array([ 1.5,  2. , -1.2,  3. ])
```

d.

$$x_1 + x_2 + 0 + x_4 = 2$$

$$2x_1 + x_2 - x_3 + x_4 = 1 \quad 4x_1 - x_2 + -2x_3 + 2x_4 = 0 \quad 3x_1 - x_2 - x_3 + 2x_4 = -3$$

```
In [32]: Fe=[[1,1,0,1,2],[2,1,-1,1,1],[4,-1,-2,2,0],[3,-1,-1,2,-3]]
          eliminacion_gaussiana(Fe)
```

```
[01-07 17:23:50][INFO]
[[ 1.  1.  0.  1.  2.]
 [ 0. -1. -1. -1. -3.]
 [ 0. -5. -2. -2. -8.]
 [ 0. -4. -1. -1. -9.]]
[01-07 17:23:50][INFO]
[[ 1.  1.  0.  1.  2.]
 [ 0. -1. -1. -1. -3.]
 [ 0.  0.  3.  3.  7.]
 [ 0.  0.  3.  3.  3.]]
[01-07 17:23:50][INFO]
[[ 1.  1.  0.  1.  2.]
 [ 0. -1. -1. -1. -3.]
 [ 0.  0.  3.  3.  7.]
 [ 0.  0.  0.  0. -4.]]
```

```
-----
ValueError                                Traceback (most recent call last)
Cell In[32], line 2
      1 Fe=[[1,1,0,1,2],[2,1,-1,1,1],[4,-1,-2,2,0],[3,-1,-1,2,-3]]
----> 2 eliminacion_gaussiana(Fe)

File c:\Users\pc\Videos\2025B\metodos_numericos\Metodos_numericos_2025B_Ulloa-Fra
ncisco\Tareas\Tarea09\tarea09_funciones.py:232, in eliminacion_gaussiana(A)
    229     logging.info(f"\n{A}")
    231     if A[n - 1, n - 1] == 0:
--> 232         raise ValueError("No existe solución única.")
    234 # -----
    235 # Sustitución hacia atrás
    236 # -----
    237 solucion = np.zeros(n)

ValueError: No existe solución única.
```

4. Use el algoritmo de eliminación gaussiana y la aritmética computacional de precisión de 32 bits para resolver los siguientes sistemas lineales.

a.

$$\frac{1}{4}x_1 + \frac{1}{5}x_2 + \frac{1}{6}x_3 = 9$$

$$\frac{1}{3}x_1 + \frac{1}{4}x_2 + \frac{1}{5}x_3 = 8 \quad \frac{1}{2}x_1 + x_2 + 2x_3 = 8$$

```
In [ ]: Gh=[[1/4,1/5,1/6,9],[1/3,1/4,1/5,8],[1/2,1,2,8]]
        eliminacion_gaussiana_redondeo32bits(Gh)
```

```
[01-07 17:00:44][INFO]
[[ 2.50000000e-01  2.00000000e-01  1.66666667e-01  9.00000000e+00]
 [-9.93410776e-09 -1.66666750e-02 -2.22222283e-02 -4.00000048e+00]
 [ 0.00000000e+00  6.00000024e-01  1.66666663e+00 -1.00000000e+01]]
[01-07 17:00:44][INFO]
[[ 2.50000000e-01  2.00000000e-01  1.66666667e-01  9.00000000e+00]
 [-9.93410776e-09 -1.66666750e-02 -2.22222283e-02 -4.00000048e+00]
 [ 0.00000000e+00 -2.13582041e-08  8.66666734e-01 -1.53999954e+02]]
```

Gauss con aritmética de 32 bits:
la solución es:

```
Out[ ]: array([-227.07678223,  476.92285156, -177.69224102])
```

$$\begin{aligned} 3.333x_1 + 15920x_2 - 10.333x_3 &= 15913, \\ \text{b. } 2.222x_1 + 16.71x_2 + 9.612x_3 &= 28.544, \\ 1.5611x_1 + 5.1791x_2 + 1.6582x_3 &= 8.4264. \end{aligned}$$

In []: `Sc=[[3.333, 15920, -10.333, 15913], [2.222, 16.71, 9.612, 28.544], [1.5611, 5.1791, 1.6582, 8.4264]]`
`eliminacion_gaussiana_redondeo32bits(Sc)`

```
[01-07 17:00:55][INFO]
[[ 1.56110000e+00  5.17910000e+00  1.65820000e+00  8.42640000e+00]
 [-2.27689743e-08  9.33830070e+00  7.25179195e+00  1.65502377e+01]
 [ 1.51944164e-07  1.59089424e+04 -1.38733120e+01  1.58950098e+04]]
[01-07 17:00:55][INFO]
[[ 1.56110000e+00  5.17910000e+00  1.65820000e+00  8.42640000e+00]
 [-2.27689743e-08  9.33830070e+00  7.25179195e+00  1.65502377e+01]
 [ 1.51944164e-07  3.63058876e-04 -1.23681914e+04 -1.23003525e+04]]
```

Gauss con aritmética de 32 bits:

la solución es:

Out[]: `array([1.02379012, 0.99999154, 0.99451505])`

$$x_1 + \frac{1}{2}x_2 + \frac{1}{3}x_3 + \frac{1}{4}x_4 = \frac{1}{6},$$

$$\frac{1}{2}x_1 + \frac{1}{3}x_2 + \frac{1}{4}x_3 + \frac{1}{5}x_4 = \frac{1}{7},$$

c.

$$\frac{1}{3}x_1 + \frac{1}{4}x_2 + \frac{1}{5}x_3 + \frac{1}{6}x_4 = \frac{1}{8},$$

$$\frac{1}{4}x_1 + \frac{1}{5}x_2 + \frac{1}{6}x_3 + \frac{1}{7}x_4 = \frac{1}{9}$$

In []: `Dc = [`
`[1, 1/2, 1/3, 1/4, 1/6],`
`[1/2, 1/3, 1/4, 1/5, 1/5],`
`[1/3, 1/4, 1/5, 1/6, 1/6],`
`[1/4, 1/5, 1/6, 1/7, 1/7]]`
`eliminacion_gaussiana_redondeo32bits(Dc)`

```
[01-07 17:02:41][INFO]
[[ 2.50000000e-01  2.00000000e-01  1.66666667e-01  1.42857143e-01
  1.42857143e-01]
 [ 0.00000000e+00 -6.66666701e-02 -8.33333358e-02 -8.57142881e-02
 -8.57142881e-02]
 [-9.93410776e-09 -1.66666750e-02 -2.22222283e-02 -2.38095298e-02
 -2.38095298e-02]
 [ 0.00000000e+00 -3.00000012e-01 -3.33333343e-01 -3.21428567e-01
 -4.04761910e-01]]
[01-07 17:02:41][INFO]
[[ 2.50000000e-01  2.00000000e-01  1.66666667e-01  1.42857143e-01
  1.42857143e-01]
 [-9.93410776e-09 -1.66666750e-02 -2.22222283e-02 -2.38095298e-02
 -2.38095298e-02]
 [ 0.00000000e+00  1.98680761e-09  5.55554032e-03  9.52379126e-03
  9.52379126e-03]
 [ 0.00000000e+00  1.06791020e-08  6.6665956e-02  1.07142791e-01
  2.38094442e-02]]
[01-07 17:02:41][INFO]
[[ 2.50000000e-01  2.00000000e-01  1.66666667e-01  1.42857143e-01
  1.42857143e-01]
 [-9.93410776e-09 -1.66666750e-02 -2.22222283e-02 -2.38095298e-02
 -2.38095298e-02]
 [ 0.00000000e+00  1.98680761e-09  5.55554032e-03  9.52379126e-03
  9.52379126e-03]
 [ 0.00000000e+00  1.06791020e-08  4.97010433e-10 -7.14289490e-03
 -9.04762447e-02]]
```

Gauss con aritmética de 32 bits:
la solución es:

```
Out[ ]: array([ -1.33333135,   9.99996758, -19.99991417,  12.6666073 ])
```

$$\begin{array}{rclclcl}
 2x_1 & +x_2 & -x_3 & +x_4 & -3x_5 & = & 7, \\
 x_1 & & +2x_3 & -x_4 & +x_5 & = & 2, \\
 \mathbf{d.} & -2x_2 & -x_3 & +x_4 & -x_5 & = & 6, \\
 3x_1 & +x_2 & -x_3 & & +5x_5 & = & 6, \\
 x_1 & -x_2 & -x_3 & x_4 & +x_5 & = & -3
 \end{array}$$

```
In [ ]: Ew = [
    [2, 1, -1, 1, -3, 7],
    [1, 0, 2, -1, 1, 2],
    [0, -2, -1, 1, -1, -5],
    [3, 1, -4, 0, 5, 6],
    [1, -1, -1, -1, 1, -3]]
eliminacion_gaussiana_redondeo32bits(Ew)
```

```
[01-07 17:11:27][INFO]
[[ 1.  0.  2. -1.  1.  2.]
 [ 0.  1. -5.  3. -5.  3.]
 [ 0. -2. -1.  1. -1. -5.]
 [ 0.  1. -10.  3.  2.  0.]
 [ 0. -1. -3.  0.  0. -5.]]
[01-07 17:11:27][INFO]
[[ 1.  0.  2. -1.  1.  2.]
 [ 0.  1. -5.  3. -5.  3.]
 [ 0.  0. -11.  7. -11.  1.]
 [ 0.  0. -5.  0.  7. -3.]
 [ 0.  0. -8.  3. -5. -2.]]
[01-07 17:11:27][INFO]
[[ 1.00000000e+00  0.00000000e+00  2.00000000e+00 -1.00000000e+00
  1.00000000e+00  2.00000000e+00]
 [ 0.00000000e+00  1.00000000e+00 -5.00000000e+00  3.00000000e+00
 -5.00000000e+00  3.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00 -5.00000000e+00  0.00000000e+00
  7.00000000e+00 -3.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  2.38418579e-07  7.00000000e+00
 -2.63999996e+01  7.60000038e+00]
 [ 0.00000000e+00  0.00000000e+00  1.19209290e-07  3.00000000e+00
 -1.62000008e+01  2.80000019e+00]]
[01-07 17:11:27][INFO]
[[ 1.00000000e+00  0.00000000e+00  2.00000000e+00 -1.00000000e+00
  1.00000000e+00  2.00000000e+00]
 [ 0.00000000e+00  1.00000000e+00 -5.00000000e+00  3.00000000e+00
 -5.00000000e+00  3.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00 -5.00000000e+00  0.00000000e+00
  7.00000000e+00 -3.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  1.19209290e-07  3.00000000e+00
 -1.62000008e+01  2.80000019e+00]
 [ 0.00000000e+00  0.00000000e+00  2.38418579e-07  2.38418579e-07
  1.14000006e+01  1.06666684e+00]]
```

Gauss con aritmética de 32 bits:

la solución es:

```
Out[ ]: array([1.88304102, 2.80701756, 0.73099416, 1.43859661, 0.09356726])
```

5. Dado el sistema lineal:

$$x_1 - x_2 + \alpha x_3 = -2$$

$$-x_1 + 2x_2 - \alpha x_3 = 3 \quad \alpha x_1 + x_2 + x_3 = 2$$

a. Encuentra el valor(es) de α para los que el sistema no tiene solución b. Encuentre los valor(es) de α para los que el sistema tiene un número infinito de soluciones.

Del sistema de ecuaciones tenemos la siguiente matriz A:

$$\begin{pmatrix} 1 & -1 & \alpha \\ -1 & 2 & -\alpha \\ \alpha & 1 & 1 \end{pmatrix}$$

Si $\det(A) = 0$, entonces el sistema tiene infinitas soluciones o ninguna. Operando:

representa un equilibrio donde existe un suministro diario de alimento para cumplir con el promedio diario de consumo de cada especie.

a. Si

$$A = [a_{ij}] = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 1 & 0 & 2 & 2 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$x = (x_j) = [1000, 500, 350, 400]$ y $b = [3500, 2700, 900]$. **¿Existe suficiente alimento para satisfacer el consumo promedio diario?**

Sí, ya que el alimento requerido es menor, al suministro diario obtenido.

b. ¿Cuál es el número máximo de animales de cada especie que se podría agregar de forma individual al sistema con el suministro de alimento qque cumpla con el consumo?

$$x_1 \leq 200, x_2 \leq 150, x_3 \leq 100, x_4 \leq 100$$

c. Si la especie 1 se extingue, ¿qué cantidad de incremento individual de las especies restantes se podría soportar?

$$x_2 \leq 650, x_3 \leq 150, x_4 \leq 150$$

d. Si la especie 2 se extingue, ¿qué cantidad de incremento individual de las especies restantes se podría soportar?

No es posible incrementar las poblaciones de las especies restantes si la especie 2 se extingue.

7. Repita el ejercicio 4 con el método Gauss_Jordan

a.

$$\frac{1}{4}x_1 + \frac{1}{5}x_2 + \frac{1}{6}x_3 = 9$$

$$\frac{1}{3}x_1 + \frac{1}{4}x_2 + \frac{1}{5}x_3 = 8 \quad \frac{1}{2}x_1 + x_2 + 2x_3 = 8$$

```
In [3]: Gh=[[1/4,1/5,1/6,9],[1/3,1/4,1/5,8],[1/2,1,2,8]]
gauss_jordan_redondeo32bits(Gh)
```

```
[01-07 17:26:05][INFO]
[[ 1.0000000e+00  8.0000001e-01  6.6666669e-01  3.6000000e+01]
 [ 0.0000000e+00 -1.6666681e-02 -2.222236e-02 -4.0000000e+00]
 [ 0.0000000e+00  6.0000002e-01  1.6666666e+00 -1.0000000e+01]]
[01-07 17:26:05][INFO]
[[ 1.          0.          -0.3999998 -155.99985  ]
 [ 0.          1.          1.3333333  239.9998  ]
 [ 0.          0.          0.8666668 -153.9999  ]]
[01-07 17:26:05][INFO]
[[ 1.          0.          0.          -227.07668]
 [ 0.          1.          0.          476.9226  ]
 [ 0.          0.          1.          -177.69215]]
```

Gauss-Jordan:

la solución es con aritmetica de 32 bits:

```
Out[3]: array([-227.07668,  476.9226 , -177.69215], dtype=float32)
```

$$\begin{aligned} 3.333x_1 + 15920x_2 - 10.333x_3 &= 15913, \\ \text{b. } 2.222x_1 + 16.71x_2 + 9.612x_3 &= 28.544, \\ 1.5611x_1 + 5.1791x_2 + 1.6582x_3 &= 8.4264. \end{aligned}$$

```
In [4]: Sc=[[3.333, 15920, -10.333, 15913], [2.222, 16.71, 9.612, 28.544], [1.5611, 5.17
gauss_jordan_redondeo32bits(Sc)
```

```
[01-07 17:26:09][INFO]
[[ 1.0000000e+00  3.3175967e+00  1.0621997e+00  5.3977323e+00]
 [ 0.0000000e+00  9.3382998e+00  7.2517929e+00  1.6550240e+01]
 [ 0.0000000e+00  1.5908942e+04 -1.3873312e+01  1.5895010e+04]]
[01-07 17:26:09][INFO]
[[ 1.0000000e+00  0.0000000e+00 -1.5141283e+00 -4.8203421e-01]
 [ 0.0000000e+00  1.0000000e+00  7.7656460e-01  1.7722969e+00]
 [ 0.0000000e+00  0.0000000e+00 -1.2368194e+04 -1.2300359e+04]]
[01-07 17:26:09][INFO]
[[1.          0.          0.          1.0237896  ]
 [0.          1.          0.          0.9999915  ]
 [0.          0.          1.          0.99451536]]
```

Gauss-Jordan:

la solución es con aritmetica de 32 bits:

```
Out[4]: array([1.0237896 , 0.9999915 , 0.99451536], dtype=float32)
```

$$\begin{aligned} x_1 + \frac{1}{2}x_2 + \frac{1}{3}x_3 + \frac{1}{4}x_4 &= \frac{1}{6}, \\ \frac{1}{2}x_1 + \frac{1}{3}x_2 + \frac{1}{4}x_3 + \frac{1}{5}x_4 &= \frac{1}{7}, \\ \text{c. } \frac{1}{3}x_1 + \frac{1}{4}x_2 + \frac{1}{5}x_3 + \frac{1}{6}x_4 &= \frac{1}{8}, \\ \frac{1}{4}x_1 + \frac{1}{5}x_2 + \frac{1}{6}x_3 + \frac{1}{7}x_4 &= \frac{1}{9} \end{aligned}$$

```
In [5]: Dc = [
[1, 1/2, 1/3, 1/4, 1/6],
[1/2, 1/3, 1/4, 1/5, 1/5],
[1/3, 1/4, 1/5, 1/6, 1/6],
```

```
[1/4, 1/5, 1/6, 1/7, 1/7]]
gauss_jordan_redondeo32bits(Dc)
```

```
[01-07 17:26:13][INFO]
[[ 1.          0.8          0.66666667  0.5714286  0.5714286 ]
 [ 0.         -0.06666666 -0.08333333 -0.0857143 -0.0857143 ]
 [ 0.         -0.01666668 -0.02222224 -0.02380954 -0.02380954]
 [ 0.         -0.3         -0.33333334 -0.3214286 -0.4047619 ]]

[01-07 17:26:13][INFO]
[[ 1.          0.         -0.39999998 -0.57142836 -0.57142836]
 [ 0.          1.          1.3333333  1.4285711  1.4285711 ]
 [ 0.          0.          0.00555552  0.00952377  0.00952377]
 [ 0.          0.          0.06666657  0.10714275  0.02380943]]

[01-07 17:26:13][INFO]
[[ 1.          0.          0.          0.1142875  0.1142875 ]
 [ 0.          1.          0.         -0.857149  -0.857149 ]
 [ 0.          0.          1.          1.7142905  1.7142905 ]
 [ 0.          0.          0.         -0.00714312 -0.09047644]]

[01-07 17:26:13][INFO]
[[ 1.          0.          0.          0.         -1.3333038]
 [ 0.          1.          0.          0.          9.999694 ]
 [ 0.          0.          1.          0.         -19.9993 ]
 [ 0.          0.          0.          1.         12.666226 ]]
```

Gauss-Jordan:

la solución es con aritmetica de 32 bits:

```
Out[5]: array([ -1.3333038,  9.999694 , -19.9993 , 12.666226 ], dtype=float32)
```

$$\begin{array}{rrrrrr}
 2x_1 & +x_2 & -x_3 & +x_4 & -3x_5 & = 7, \\
 x_1 & & +2x_3 & -x_4 & +x_5 & = 2, \\
 \mathbf{d.} & -2x_2 & -x_3 & +x_4 & -x_5 & = 6, \\
 3x_1 & +x_2 & -x_3 & & +5x_5 & = 6, \\
 x_1 & -x_2 & -x_3 & x_4 & +x_5 & = -3
 \end{array}$$

```
In [6]: Ew = [
[2, 1, -1, 1, -3, 7],
[1, 0, 2, -1, 1, 2],
[0, -2, -1, 1, -1, -5],
[3, 1, -4, 0, 5, 6],
[1, -1, -1, -1, 1, -3]]
gauss_jordan_redondeo32bits(Ew)
```

```

[01-07 17:26:19][INFO]
[[ 1.  0.  2. -1.  1.  2.]
 [ 0.  1. -5.  3. -5.  3.]
 [ 0. -2. -1.  1. -1. -5.]
 [ 0.  1. -10.  3.  2.  0.]
 [ 0. -1. -3.  0.  0. -5.]]
[01-07 17:26:19][INFO]
[[ 1.  0.  2. -1.  1.  2.]
 [ 0.  1. -5.  3. -5.  3.]
 [ 0.  0. -11.  7. -11.  1.]
 [ 0.  0. -5.  0.  7. -3.]
 [ 0.  0. -8.  3. -5. -2.]]
[01-07 17:26:19][INFO]
[[ 1.          0.          0.          -1.          3.8
  0.79999995]
 [ 0.          1.          0.          3.          -12.
  6.          ]
 [ 0.          0.          1.          -0.          -1.4
  0.6          ]
 [ 0.          0.          0.          7.          -26.4
  7.6000004 ]
 [ 0.          0.          0.          3.          -16.2
  2.8000002 ]]
[01-07 17:26:19][INFO]
[[ 1.          0.          0.          0.          -1.6000001  1.7333333]
 [ 0.          1.          0.          0.          4.200001  3.1999998]
 [ 0.          0.          1.          0.          -1.4      0.6      ]
 [ 0.          0.          0.          1.          -5.4      0.9333334]
 [ 0.          0.          0.          0.          11.4      1.0666666]]
[01-07 17:26:19][INFO]
[[1.          0.          0.          0.          0.          1.8830409 ]
 [0.          1.          0.          0.          0.          2.8070173 ]
 [0.          0.          1.          0.          0.          0.73099416]
 [0.          0.          0.          1.          0.          1.4385965 ]
 [0.          0.          0.          0.          1.          0.09356725]]

```

Gauss-Jordan:

la solución es con aritmetica de 32 bits:

```

Out[6]: array([1.8830409 , 2.8070173 , 0.73099416, 1.4385965 , 0.09356725],
              dtype=float32)

```