Taller 2b

Nombre: David Alejandro Díaz Pineda

Haga modificaciones a las funciones base del siguiente código: https://github.com/ztjona/MN-prueba-02/tree/main

In [1]:
```
%load_ext autoreload
```

In [2]:
```
# ---------------------------- logging -------------------------
import logging
from sys import stdout
from datetime import datetime

logging.basicConfig(
    level=logging.INFO,
    format="[%(asctime)s][%(levelname)s] %(message)s",
    stream=stdout,
    datefmt="%m-%d %H:%M:%S",
)
logging.info(datetime.now())
```

```
[01-13 19:34:08][INFO] 2026-01-13 19:34:08.563550
```

In [3]:
```
%autoreload 2
from src import eliminacion_gaussiana
```

```
[01-13 19:34:08][INFO] 2026-01-13 19:34:08.677818
```

In [4]:
```
import numpy as np


# ####################################################################
def gauss_jordan(A: np.ndarray) -> np.ndarray:
    """Realiza la eliminación de Gauss-Jordan

    ## Parameters

    ``A``: matriz del sistema de ecuaciones lineales. Debe ser de tamaño n-by-(k

    ## Return

    ``A``: matriz reducida por filas.

    """
    if not isinstance(A, np.ndarray):
        logging.debug("Convirtiendo A a numpy array.")
        A = np.array(A, dtype=float)
    n = A.shape[0]

    for i in range(0, n):  # loop por columna

        # --- encontrar pivote
        p = None  # default, first element
        for pi in range(i, n):
            if A[pi, i] == 0:
```

```python
                # must be nonzero
                continue

            if p is None:
                # first nonzero element
                p = pi
                continue

            if abs(A[pi, i]) < abs(A[p, i]):
                p = pi

        if p is None:
            # no pivot found.
            raise ValueError("No existe solución única.")

        if p != i:
            # swap rows
            logging.debug(f"Intercambiando filas {i} y {p}")
            _aux = A[i, :].copy()
            A[i, :] = A[p, :].copy()
            A[p, :] = _aux

        # --- Eliminación: loop por fila
        # for j in range(i + 1, n): # Eliminación gaussiana
        for j in range(n):  # Gauss-Jordan
            if j == i:
                continue  # skip pivot row

            m = A[j, i] / A[i, i]
            A[j, i:] = A[j, i:] - m * A[i, i:]

            # dividir para la diagonal
        A[i, :] = A[i, :] / A[i, i]

        logging.info(f"\n{A}")

    if A[n - 1, n - 1] == 0:
        raise ValueError("No existe solución única.")

        print(f"\n{A}")
    # # --- Sustitución hacia atrás
    # solucion = np.zeros(n)
    # solucion[n - 1] = A[n - 1, n] / A[n - 1, n - 1]

    # for i in range(n - 2, -1, -1):
    #     suma = 0
    #     for j in range(i + 1, n):
    #         suma += A[i, j] * solucion[j]
    #     solucion[i] = (A[i, n] - suma) / A[i, i]

    return A
```

In [5]:

```python
def matriz_inversa(A: np.ndarray) -> np.ndarray:
    """Calcula la matriz inversa de A usando eliminación de Gauss-Jordan

    ## Parameters

    ``A``: matriz cuadrada a invertir.

    ## Return
```

```
    ``A_inv``: matriz inversa de A.

    """
    if not isinstance(A, np.ndarray):
        logging.debug("Convirtiendo A a numpy array.")
        A = np.array(A, dtype=float)

    n = A.shape[0]

    A_aumentada = np.hstack((A, np.eye(n)))

    A_reducida = gauss_jordan(A_aumentada)

    A_inversa = A_reducida[:, n:]

    return A_inversa
```

1) Para encontrar la matriz inversa de las siguientes matrices:

$$\begin{bmatrix} 1 & 3 & 4 \\ 2 & 1 & 3 \\ 4 & 2 & 1 \end{bmatrix}$$

In [6]:
```
A1=[[1,3,4],
   [2,1,3],
   [4,2,1]
   ]
```

In [7]:
```
matriz_inversa(np.array(A1))
```

```
[01-13 19:34:09][INFO]
[[  1.    3.    4.    1.    0.    0.]
 [  0.   -5.   -5.   -2.    1.    0.]
 [  0.  -10.  -15.   -4.    0.    1.]]
[01-13 19:34:09][INFO]
[[ 1.    0.    1.   -0.2   0.6   0. ]
 [-0.    1.    1.    0.4  -0.2  -0. ]
 [ 0.    0.   -5.    0.   -2.    1. ]]
[01-13 19:34:09][INFO]
[[ 1.    0.    0.   -0.2   0.2   0.2]
 [-0.    1.    0.    0.4  -0.6   0.2]
 [-0.   -0.    1.   -0.    0.4  -0.2]]
```

Out[7]:
```
array([[-0.2,  0.2,  0.2],
       [ 0.4, -0.6,  0.2],
       [-0. ,  0.4, -0.2]])
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 4 \\ 5 & 6 & 0 \end{bmatrix}$$

In [8]:
```
A2 = [[1,2,3],
      [0,1,4],
```

```
        [5,6,0]]
```

In [9]:
```
matriz_inversa(np.array(A2))
```

```
[01-13 19:34:09][INFO]
[[  1.    2.    3.    1.    0.    0.]
 [  0.    1.    4.    0.    1.    0.]
 [  0.   -4.  -15.   -5.    0.    1.]]
[01-13 19:34:09][INFO]
[[ 1.   0.  -5.   1.  -2.   0.]
 [ 0.   1.   4.   0.   1.   0.]
 [ 0.   0.   1.  -5.   4.   1.]]
[01-13 19:34:09][INFO]
[[  1.    0.    0.  -24.   18.    5.]
 [  0.    1.    0.   20.  -15.   -4.]
 [  0.    0.    1.   -5.    4.    1.]]
```

Out[9]:
```
array([[-24.,  18.,   5.],
       [ 20., -15.,  -4.],
       [ -5.,   4.,   1.]])
```

$$\begin{bmatrix} 4 & 2 & 1 \\ 2 & 1 & 3 \\ 1 & 3 & 4 \end{bmatrix}$$

In [10]:
```
A3= [[4,2,1],
     [2,1,3],
     [1,3,4]
    ]
```

In [11]:
```
matriz_inversa(np.array(A3))
```

```
[01-13 19:34:09][INFO]
[[  1.    3.    4.    0.    0.    1.]
 [  0.   -5.   -5.    0.    1.   -2.]
 [  0.  -10.  -15.    1.    0.   -4.]]
[01-13 19:34:09][INFO]
[[ 1.    0.    1.    0.    0.6 -0.2]
 [-0.    1.    1.   -0.   -0.2  0.4]
 [ 0.    0.   -5.    1.   -2.    0. ]]
[01-13 19:34:09][INFO]
[[ 1.    0.    0.    0.2  0.2 -0.2]
 [-0.    1.    0.    0.2 -0.6  0.4]
 [-0.   -0.    1.   -0.2  0.4 -0. ]]
```

Out[11]:
```
array([[ 0.2,  0.2, -0.2],
       [ 0.2, -0.6,  0.4],
       [-0.2,  0.4, -0. ]])
```

$$\begin{bmatrix} 2 & 4 & 6 & 1 \\ 4 & 7 & 5 & -6 \\ 2 & 5 & 18 & 10 \\ 6 & 12 & 38 & 16 \end{bmatrix}$$

In [12]:
```
A4 = [[2,4,6,1],
      [4,7,5,-6],
```

```
      [2,5,18,10],
      [6,12,38,16]]
```

In [13]:
```
matriz_inversa(np.array(A4))
```

```
[01-13 19:34:09][INFO]
[[ 1.   2.   3.   0.5  0.5  0.   0.   0. ]
 [ 0.  -1.  -7.  -8.  -2.   1.   0.   0. ]
 [ 0.   1.  12.   9.  -1.   0.   1.   0. ]
 [ 0.   0.  20.  13.  -3.   0.   0.   1. ]]
[01-13 19:34:09][INFO]
[[  1.    0.  -11.  -15.5  -3.5   2.    0.    0. ]
 [ -0.    1.    7.    8.    2.   -1.   -0.   -0. ]
 [  0.    0.    5.    1.   -3.    1.    1.    0. ]
 [  0.    0.   20.   13.   -3.    0.    0.    1. ]]
[01-13 19:34:09][INFO]
[[  1.    0.    0.  -13.3 -10.1   4.2   2.2   0. ]
 [ -0.    1.    0.    6.6   6.2  -2.4  -1.4  -0. ]
 [  0.    0.    1.    0.2  -0.6   0.2   0.2   0. ]
 [  0.    0.    0.    9.    9.   -4.   -4.    1. ]]
[01-13 19:34:09][INFO]
[[ 1.          0.          0.          0.          3.2        -1.71111111
  -3.71111111  1.47777778]
 [-0.          1.          0.          0.         -0.4         0.53333333
   1.53333333 -0.73333333]
 [ 0.          0.          1.          0.         -0.8         0.28888889
   0.28888889 -0.02222222]
 [ 0.          0.          0.          1.          1.         -0.44444444
  -0.44444444  0.11111111]]
```

Out[13]:
```
array([[ 3.2       , -1.71111111, -3.71111111,  1.47777778],
       [-0.4       ,  0.53333333,  1.53333333, -0.73333333],
       [-0.8       ,  0.28888889,  0.28888889, -0.02222222],
       [ 1.       , -0.44444444, -0.44444444,  0.11111111]])
```

2) Calcule la descomposición LU para estas matrices y encuentre la solución para estos vectores de valores independientes b

In [14]:
```
from src import descomposicion_LU, resolver_LU
```

$$b1 = \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix}$$

In [15]:
```
L1, U1 = descomposicion_LU(np.array(A1))
b1 = np.array([1,2,4])
x1 = resolver_LU(L1, U1, b1)
print("Matriz L:")
print(L1)
print("Matriz U:")
print(U1)
print("Solución del sistema Ax=b:")
print(x1)
```

```
[01-13 19:34:09][INFO]
[[  1.    3.    4.]
 [  0.   -5.   -5.]
 [  0. -10. -15.]]
[01-13 19:34:09][INFO]
[[ 1.  3.  4.]
 [ 0. -5. -5.]
 [ 0.  0. -5.]]
[01-13 19:34:09][INFO]
[[ 1.  3.  4.]
 [ 0. -5. -5.]
 [ 0.  0. -5.]]
[01-13 19:34:09][INFO] Sustitución hacia adelante
[01-13 19:34:09][INFO] y =
[[1.]
 [0.]
 [0.]]
[01-13 19:34:09][INFO] Sustitución hacia atrás
[01-13 19:34:09][INFO] i = 1
[01-13 19:34:09][INFO] suma = [0.]
[01-13 19:34:09][INFO] U[i, i] = -5.0
[01-13 19:34:09][INFO] y[i] = [0.]
[01-13 19:34:09][INFO] i = 0
[01-13 19:34:09][INFO] suma = [0.]
[01-13 19:34:09][INFO] U[i, i] = 1.0
[01-13 19:34:09][INFO] y[i] = [1.]
Matriz L:
[[1. 0. 0.]
 [2. 1. 0.]
 [4. 2. 1.]]
Matriz U:
[[ 1.  3.  4.]
 [ 0. -5. -5.]
 [ 0.  0. -5.]]
Solución del sistema Ax=b:
[[ 1.]
 [-0.]
 [-0.]]
```

$$b2 = \begin{bmatrix} 3 \\ -5 \\ 2 \end{bmatrix}$$

In [16]:

```
L2, U2 = descomposicion_LU(np.array(A2))
b2 = np.array([3,-5,2])
x2 = resolver_LU(L2, U2, b2)
print("Matriz L:")
print(L2)
print("Matriz U:")
print(U2)
print("Solución del sistema Ax=b:")
print(x2)
```

```
[01-13 19:34:09][INFO]
[[  1.   2.   3.]
 [  0.   1.   4.]
 [  0.  -4. -15.]]
[01-13 19:34:09][INFO]
[[1. 2. 3.]
 [0. 1. 4.]
 [0. 0. 1.]]
[01-13 19:34:09][INFO]
[[1. 2. 3.]
 [0. 1. 4.]
 [0. 0. 1.]]
[01-13 19:34:09][INFO] Sustitución hacia adelante
[01-13 19:34:09][INFO] y =
[[  3.]
 [ -5.]
 [-33.]]
[01-13 19:34:09][INFO] Sustitución hacia atrás
[01-13 19:34:09][INFO] i = 1
[01-13 19:34:09][INFO] suma = [-132.]
[01-13 19:34:09][INFO] U[i, i] = 1.0
[01-13 19:34:09][INFO] y[i] = [-5.]
[01-13 19:34:09][INFO] i = 0
[01-13 19:34:09][INFO] suma = [155.]
[01-13 19:34:09][INFO] U[i, i] = 1.0
[01-13 19:34:09][INFO] y[i] = [3.]
Matriz L:
[[ 1.  0.  0.]
 [ 0.  1.  0.]
 [ 5. -4.  1.]]
Matriz U:
[[1. 2. 3.]
 [0. 1. 4.]
 [0. 0. 1.]]
Solución del sistema Ax=b:
[[-152.]
 [ 127.]
 [ -33.]]
```

$$b3 = \begin{bmatrix} 7 \\ 8 \\ -1 \end{bmatrix}$$

In [17]:
```python
L3, U3 = descomposicion_LU(np.array(A3))
b3 = np.array([7,8,-1])
x3 = resolver_LU(L3, U3, b3)
print("Matriz L:")
print(L3)
print("Matriz U:")
print(U3)
print("Solución del sistema Ax=b:")
print(x3)
```

```
[01-13 19:34:09][INFO]
[[4.   2.   1.  ]
 [0.   0.   2.5 ]
 [0.   2.5  3.75]]
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[17], line 1
----> 1 L3, U3 = descomposicion_LU(np.array(A3))
      2 b3 = np.array([7,8,-1])
      3 x3 = resolver_LU(L3, U3, b3)

File D:\EPN\Cuarto Semestre\Métodos Numéricos\Taller2b\taller_LU\src\linear_syst_
methods.py:127, in descomposicion_LU(A)
    123 for i in range(0, n):  # loop por columna
    124 
    125     # --- determinar pivote
    126     if A[i, i] == 0:
--> 127         raise ValueError("No existe solución única.")
    129     # --- Eliminación: loop por fila
    130     L[i, i] = 1

ValueError: No existe solución única.
```

$$b1 = \begin{bmatrix} 1 \\ 2 \\ 4 \\ 5 \end{bmatrix}$$

In [18]:
```
L4, U4 = descomposicion_LU(np.array(A4))
b4 = np.array([1,2,4,5])
x4 = resolver_LU(L4, U4, b4)
print("Matriz L:")
print(L4)
print("Matriz U:")
print(U4)
print("Solución del sistema Ax=b:")
print(x4)
```

```
[01-13 19:34:29][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  1. 12.  9.]
 [ 0.  0. 20. 13.]]
[01-13 19:34:29][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0. 20. 13.]]
[01-13 19:34:29][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 19:34:29][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 19:34:29][INFO] Sustitución hacia adelante
[01-13 19:34:29][INFO] y =
[[  1.]
 [  0.]
 [  3.]
 [-10.]]
[01-13 19:34:29][INFO] Sustitución hacia atrás
[01-13 19:34:29][INFO] i = 2
[01-13 19:34:29][INFO] suma = [-1.11111111]
[01-13 19:34:29][INFO] U[i, i] = 5.0
[01-13 19:34:29][INFO] y[i] = [3.]
[01-13 19:34:29][INFO] i = 1
[01-13 19:34:29][INFO] suma = [3.13333333]
[01-13 19:34:29][INFO] U[i, i] = -1.0
[01-13 19:34:29][INFO] y[i] = [0.]
[01-13 19:34:29][INFO] i = 0
[01-13 19:34:29][INFO] suma = [16.35555556]
[01-13 19:34:29][INFO] U[i, i] = 2.0
[01-13 19:34:29][INFO] y[i] = [1.]
Matriz L:
[[ 1.  0.  0.  0.]
 [ 2.  1.  0.  0.]
 [ 1. -1.  1.  0.]
 [ 3. -0.  4.  1.]]
Matriz U:
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
Solución del sistema Ax=b:
[[-7.67777778]
 [ 3.13333333]
 [ 0.82222222]
 [-1.11111111]]
```

$$b2 = \begin{bmatrix} 3 \\ -5 \\ 2 \\ 6 \end{bmatrix}$$

In [19]:

```
L5, U5 = descomposicion_LU(np.array(A4))
b5 = np.array([3,-5,2,6])
x5 = resolver_LU(L5, U5, b5)
print("Matriz L:")
print(L5)
print("Matriz U:")
print(U5)
print("Solución del sistema Ax=b:")
print(x5)
```

```
[01-13 19:34:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  1. 12.  9.]
 [ 0.  0. 20. 13.]]
[01-13 19:34:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0. 20. 13.]]
[01-13 19:34:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 19:34:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 19:34:32][INFO] Sustitución hacia adelante
[01-13 19:34:32][INFO] y =
[[  3.]
 [-11.]
 [-12.]
 [ 45.]]
[01-13 19:34:32][INFO] Sustitución hacia atrás
[01-13 19:34:32][INFO] i = 2
[01-13 19:34:32][INFO] suma = [5.]
[01-13 19:34:32][INFO] U[i, i] = 5.0
[01-13 19:34:32][INFO] y[i] = [-12.]
[01-13 19:34:32][INFO] i = 1
[01-13 19:34:32][INFO] suma = [-16.2]
[01-13 19:34:32][INFO] U[i, i] = -1.0
[01-13 19:34:32][INFO] y[i] = [-11.]
[01-13 19:34:32][INFO] i = 0
[01-13 19:34:32][INFO] suma = [-36.2]
[01-13 19:34:32][INFO] U[i, i] = 2.0
[01-13 19:34:32][INFO] y[i] = [3.]
Matriz L:
[[ 1.  0.  0.  0.]
 [ 2.  1.  0.  0.]
 [ 1. -1.  1.  0.]
 [ 3. -0.  4.  1.]]
Matriz U:
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
Solución del sistema Ax=b:
[[19.6]
 [-5.2]
 [-3.4]
 [ 5. ]]
```

$$b1 = \begin{bmatrix} 7 \\ 8 \\ -1 \\ 0 \end{bmatrix}$$

In [20]:

```
L5, U5 = descomposicion_LU(np.array(A4))
b5 = np.array([7,8,-1,0])
x5 = resolver_LU(L5, U5, b5)
print("Matriz L:")
print(L5)
print("Matriz U:")
print(U5)
print("Solución del sistema Ax=b:")
print(x5)
```

```
[01-13 19:34:34][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  1. 12.  9.]
 [ 0.  0. 20. 13.]]
[01-13 19:34:34][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0. 20. 13.]]
[01-13 19:34:34][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 19:34:34][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 19:34:34][INFO] Sustitución hacia adelante
[01-13 19:34:34][INFO] y =
[[  7.]
 [ -6.]
 [-14.]
 [ 35.]]
[01-13 19:34:34][INFO] Sustitución hacia atrás
[01-13 19:34:34][INFO] i = 2
[01-13 19:34:34][INFO] suma = [3.88888889]
[01-13 19:34:34][INFO] U[i, i] = 5.0
[01-13 19:34:34][INFO] y[i] = [-14.]
[01-13 19:34:34][INFO] i = 1
[01-13 19:34:34][INFO] suma = [-6.06666667]
[01-13 19:34:34][INFO] U[i, i] = -1.0
[01-13 19:34:34][INFO] y[i] = [-6.]
[01-13 19:34:34][INFO] i = 0
[01-13 19:34:34][INFO] suma = [-17.84444444]
[01-13 19:34:34][INFO] U[i, i] = 2.0
[01-13 19:34:34][INFO] y[i] = [7.]
Matriz L:
[[ 1.  0.  0.  0.]
 [ 2.  1.  0.  0.]
 [ 1. -1.  1.  0.]
 [ 3. -0.  4.  1.]]
Matriz U:
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
Solución del sistema Ax=b:
[[12.42222222]
 [-0.06666667]
 [-3.57777778]
 [ 3.88888889]]
```