Taller3B_LeninAmangandi

January 13, 2026

1 Escuela Politécnica Nacional

1.1 Métodos Numéricos

Nombre: Lenin Amangandi

Tema: Descomposición LU

Link al repositorio Taller N3B

1.2 1) Haga modificaciones a las funciones base del siguiente código: https://github.com/ztjona/MN-prueba-02/tree/main para encontrar la matriz inversa de las siguientes matrices:

[1]:

```
%load_ext autoreload
```

[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 21:07:39][INFO] 2026-01-13 21:07:39.031321
```

1.3 Matriz A_1

[3]:

```
%autoreload 2
from src import eliminacion_gaussiana
```

```
[01-13 21:07:39][INFO] 2026-01-13 21:07:39.061915
```

[4]:

```
A = [[1, 3, 4], [2, 1, 3], [4, 2, 1]]
```

[5]:

```
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), donde n es el número de incógnitas.

    ## 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:
                # 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
```

[6]:

```
from pprint import pprint

pprint(A)
gauss_jordan(A)
```

```
[[1, 3, 4], [2, 1, 3], [4, 2, 1]]
[01-13 21:07:39][INFO]
[[  1.   3.   4.]
 [  0.  -5.  -5.]
 [  0. -10. -15.]]
[01-13 21:07:39][INFO]
[[ 1.  0.  1.]
 [-0.  1.  1.]
 [ 0.  0. -5.]]
[01-13 21:07:39][INFO]
```

```
[[ 1.  0.  0.]
 [-0.  1.  0.]
 [-0. -0.  1.]]
```

[6]:
```
array([[ 1.,  0.,  0.],
       [-0.,  1.,  0.],
       [-0., -0.,  1.]])
```

[7]:
```
# Poner matriz identidad a la derecha
n = len(A)
A_aug = np.hstack((A, np.eye(n)))
A_aug
```

[7]:
```
array([[1., 3., 4., 1., 0., 0.],
       [2., 1., 3., 0., 1., 0.],
       [4., 2., 1., 0., 0., 1.]])
```

[8]:
```
np.vstack((A, np.eye(n)))
```

[8]:
```
array([[1., 3., 4.],
       [2., 1., 3.],
       [4., 2., 1.],
       [1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

[9]:
```
_m_inv = gauss_jordan(A_aug)
_m_inv
```

```
[01-13 21:07:40][INFO]
[[  1.   3.   4.   1.   0.   0.]
 [  0.  -5.  -5.  -2.   1.   0.]
 [  0. -10. -15.  -4.   0.   1.]]
[01-13 21:07:40][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 21:07:40][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]]
```

[9]:
```
array([[ 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]])
```

[10]:
```
_m_inv[:, n:]
```

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

```
[11]: np.linalg.inv(np.array(A))
```

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

1.4 Matriz A_2

```
[12]: A_2 = [[1, 2, 3], [0, 1, 4], [5, 6, 0]]
```

```
[13]: from pprint import pprint

      pprint(A_2)
      gauss_jordan(A_2)
```

```
[[1, 2, 3], [0, 1, 4], [5, 6, 0]]
[01-13 21:07:40][INFO]
[[  1.   2.   3.]
 [  0.   1.   4.]
 [  0.  -4. -15.]]
[01-13 21:07:40][INFO]
[[ 1.  0. -5.]
 [ 0.  1.  4.]
 [ 0.  0.  1.]]
[01-13 21:07:40][INFO]
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

```
[13]: array([[1., 0., 0.],
             [0., 1., 0.],
             [0., 0., 1.]])
```

```
[14]: n = len(A_2)
      A_aug = np.hstack((A_2, np.eye(n)))
      A_aug
```

```
[14]: array([[1., 2., 3., 1., 0., 0.],
             [0., 1., 4., 0., 1., 0.],
             [5., 6., 0., 0., 0., 1.]])
```

```
[15]: np.vstack((A_2, np.eye(n)))
```

```
[15]: array([[1., 2., 3.],
             [0., 1., 4.],
             [5., 6., 0.],
             [1., 0., 0.],
             [0., 1., 0.],
             [0., 0., 1.]])
```

```
[16]: _m_inv = gauss_jordan(A_aug)
      _m_inv
```

```
[01-13 21:07:40][INFO]
[[  1.   2.   3.   1.   0.   0.]
 [  0.   1.   4.   0.   1.   0.]
 [  0.  -4. -15.  -5.   0.   1.]]
[01-13 21:07:40][INFO]
[[ 1.  0. -5.  1. -2.  0.]
 [ 0.  1.  4.  0.  1.  0.]
 [ 0.  0.  1. -5.  4.  1.]]
[01-13 21:07:40][INFO]
[[  1.   0.   0. -24.  18.   5.]
 [  0.   1.   0.  20. -15.  -4.]
 [  0.   0.   1.  -5.   4.   1.]]
```

```
[16]: array([[  1.,   0.,   0., -24.,  18.,   5.],
             [  0.,   1.,   0.,  20., -15.,  -4.],
             [  0.,   0.,   1.,  -5.,   4.,   1.]])
```

```
[17]: _m_inv[:, n:]
```

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

```
[18]: np.linalg.inv(np.array(A_2))
```

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

1.5 Matriz A_3

```
[19]: A_3 = [[4, 2, 1], [2, 1, 3], [1, 3, 4]]

      from pprint import pprint

      pprint(A_3)
      gauss_jordan(A_3)
```

```
n = len(A_3)
A_aug = np.hstack((A_3, np.eye(n)))
A_aug

np.vstack((A_3, np.eye(n)))

_m_inv = gauss_jordan(A_aug)
_m_inv

_m_inv[:, n:]
```

```
[[4, 2, 1], [2, 1, 3], [1, 3, 4]]
[01-13 21:07:41][INFO]
[[  1.   3.   4.]
 [  0.  -5.  -5.]
 [  0. -10. -15.]]
[01-13 21:07:41][INFO]
[[ 1.  0.  1.]
 [-0.  1.  1.]
 [ 0.  0. -5.]]
[01-13 21:07:41][INFO]
[[ 1.  0.  0.]
 [-0.  1.  0.]
 [-0. -0.  1.]]
[01-13 21:07:41][INFO]
[[  1.   3.   4.   0.   0.   1.]
 [  0.  -5.  -5.   0.   1.  -2.]
 [  0. -10. -15.   1.   0.  -4.]]
[01-13 21:07:41][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 21:07:41][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. ]]
```

[19]:

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

[20]:

```
np.linalg.inv(np.array(A_3))
```

[20]:

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

1.6 Matriz A_4

[21]:
```
A_4 = [[2, 4, 6, 1], [4,7,5, -6], [2, 5, 18,10], [6,12, 38, 16]]

from pprint import pprint

pprint(A_4)
gauss_jordan(A_4)

n = len(A_4)
A_aug = np.hstack((A_4, np.eye(n)))
A_aug

np.vstack((A_4, np.eye(n)))

_m_inv = gauss_jordan(A_aug)
_m_inv

_m_inv[:, n:]
```

```
[[2, 4, 6, 1], [4, 7, 5, -6], [2, 5, 18, 10], [6, 12, 38, 16]]
[01-13 21:07:41][INFO]
[[ 1.   2.   3.   0.5]
 [ 0.  -1.  -7.  -8. ]
 [ 0.   1.  12.   9. ]
 [ 0.   0.  20.  13. ]]
[01-13 21:07:41][INFO]
[[  1.    0.  -11.  -15.5]
 [ -0.    1.    7.    8. ]
 [  0.    0.    5.    1. ]
 [  0.    0.   20.   13. ]]
[01-13 21:07:41][INFO]
[[  1.    0.    0.  -13.3]
 [ -0.    1.    0.    6.6]
 [  0.    0.    1.    0.2]
 [  0.    0.    0.    9. ]]
[01-13 21:07:41][INFO]
[[ 1.  0.  0.  0.]
 [-0.  1.  0.  0.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]
[01-13 21:07:41][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 21:07:41][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 21:07:41][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 21:07:41][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]]
```

```
[21]: 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]])
```

```
[22]: np.linalg.inv(np.array(A_4))
```

```
[22]: 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]])
```

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

B1)

```
[41]: from src.linear_syst_methods import descomposicion_LU, resolver_LU,␣
        ↪matriz_aumentada
      import numpy as np
      A1 = [[1, 3, 4], [2, 1, 3], [4, 2, 1]]



      b1 = [1, 2, 4]



      Ab1 = matriz_aumentada(A1, b1)

      L, U = descomposicion_LU(A1)
```

```
print("Matriz L:")
print(L)

print("\nMatriz U:")
print(U)


solucion = resolver_LU(L, U, b1)


print("\nSolución del sistema (vector X):")
print(solucion)
```

```
[01-13 21:33:11][INFO]
[[  1.   3.   4.]
 [  0.  -5.  -5.]
 [  0. -10. -15.]]
[01-13 21:33:11][INFO]
[[ 1.  3.  4.]
 [ 0. -5. -5.]
 [ 0.  0. -5.]]
[01-13 21:33:11][INFO]
[[ 1.  3.  4.]
 [ 0. -5. -5.]
 [ 0.  0. -5.]]
Matriz L:
[[1. 0. 0.]
 [2. 1. 0.]
 [4. 2. 1.]]

Matriz U:
[[ 1.  3.  4.]
 [ 0. -5. -5.]
 [ 0.  0. -5.]]
[01-13 21:33:11][INFO] Sustitución hacia adelante
[01-13 21:33:11][INFO] y =
[[1.]
 [0.]
 [0.]]
[01-13 21:33:11][INFO] Sustitución hacia atrás
[01-13 21:33:11][INFO] i = 1
[01-13 21:33:11][INFO] suma = [0.]
[01-13 21:33:11][INFO]
[[ 1.  3.  4.]
 [ 0. -5. -5.]
 [ 0.  0. -5.]]
[01-13 21:33:11][INFO]
```

```
[[ 1.  3.  4.]
 [ 0. -5. -5.]
 [ 0.  0. -5.]]
Matriz L:
[[1. 0. 0.]
 [2. 1. 0.]
 [4. 2. 1.]]

Matriz U:
[[ 1.  3.  4.]
 [ 0. -5. -5.]
 [ 0.  0. -5.]]
[01-13 21:33:11][INFO] Sustitución hacia adelante
[01-13 21:33:11][INFO] y =
[[1.]
 [0.]
 [0.]]
[01-13 21:33:11][INFO] Sustitución hacia atrás
[01-13 21:33:11][INFO] i = 1
[01-13 21:33:11][INFO] suma = [0.]
[01-13 21:33:11][INFO] U[i, i] = -5.0
[01-13 21:33:11][INFO] y[i] = [0.]
[01-13 21:33:11][INFO] i = 0
[01-13 21:33:11][INFO] suma = [0.]
[01-13 21:33:11][INFO] U[i, i] = 1.0
[01-13 21:33:11][INFO] y[i] = [1.]

Solución del sistema (vector X):
[[ 1.]
 [-0.]
 [-0.]]
```

B2)

[31]:

```
from src.linear_syst_methods import descomposicion_LU, resolver_LU,␣
  ↪matriz_aumentada

import numpy as np


A2 = [[1, 2, 3], [0, 1, 4], [5, 6, 0]]
b2 = [3, -5, 2]


Ab2 = matriz_aumentada(A2, b2)


L2, U2 = descomposicion_LU(A2)
```

```python
print("Matriz L:")
print(L2)

print("\nMatriz U:")
print(U2)

solucion2 = resolver_LU(L2, U2, b2)

print("\nSolución del sistema (vector X):")
print(solucion2)
```

```
[01-13 21:25:19][INFO]
[[  1.   2.   3.]
 [  0.   1.   4.]
 [  0.  -4. -15.]]
[01-13 21:25:19][INFO]
[[1. 2. 3.]
 [0. 1. 4.]
 [0. 0. 1.]]
[01-13 21:25:19][INFO]
[[1. 2. 3.]
 [0. 1. 4.]
 [0. 0. 1.]]
Matriz L:
[[ 1.  0.  0.]
 [ 0.  1.  0.]
 [ 5. -4.  1.]]

Matriz U:
[[1. 2. 3.]
 [0. 1. 4.]
 [0. 0. 1.]]
[01-13 21:25:19][INFO] Sustitución hacia adelante
[01-13 21:25:19][INFO] y =
[[  3.]
 [ -5.]
 [-33.]]
[01-13 21:25:19][INFO] Sustitución hacia atrás
[01-13 21:25:19][INFO] i = 1
[01-13 21:25:19][INFO] suma = [-132.]
[01-13 21:25:19][INFO]
[[1. 2. 3.]
 [0. 1. 4.]
 [0. 0. 1.]]
[01-13 21:25:19][INFO]
[[1. 2. 3.]
```

```
 [0. 1. 4.]
 [0. 0. 1.]]
Matriz L:
[[ 1.  0.  0.]
 [ 0.  1.  0.]
 [ 5. -4.  1.]]

Matriz U:
[[1. 2. 3.]
 [0. 1. 4.]
 [0. 0. 1.]]
[01-13 21:25:19][INFO] Sustitución hacia adelante
[01-13 21:25:19][INFO] y =
[[  3.]
 [ -5.]
 [-33.]]
[01-13 21:25:19][INFO] Sustitución hacia atrás
[01-13 21:25:19][INFO] i = 1
[01-13 21:25:19][INFO] suma = [-132.]
[01-13 21:25:19][INFO] U[i, i] = 1.0
[01-13 21:25:19][INFO] y[i] = [-5.]
[01-13 21:25:19][INFO] i = 0
[01-13 21:25:19][INFO] suma = [155.]
[01-13 21:25:19][INFO] U[i, i] = 1.0
[01-13 21:25:19][INFO] y[i] = [3.]

Solución del sistema (vector X):
[[-152.]
 [ 127.]
 [ -33.]]
```

B3)

[45]:

```python
from src.linear_syst_methods import descomposicion_LU, resolver_LU,␣
 ↪matriz_aumentada
import numpy as np

A3 = np.array([
    [4, 2, 1],
    [1, 3, 4],
    [2, 1, 3]
], dtype=float)

b3 = np.array([7, -1, 8], dtype=float).reshape(-1, 1)

print("Descomposición LU")
L3, U3 = descomposicion_LU(A3)
```

```python
print("Matriz L:")
print(L3)

print("\nMatriz U:")
print(U3)

solucion3 = resolver_LU(L3, U3, b3)

print("\nSolución del sistema (vector X):")
print(solucion3)
```

```
Descomposición LU
[01-13 21:44:01][INFO]
[[4.   2.   1.  ]
 [0.   2.5  3.75]
 [0.   0.   2.5 ]]
[01-13 21:44:01][INFO]
[[4.   2.   1.  ]
 [0.   2.5  3.75]
 [0.   0.   2.5 ]]
[01-13 21:44:01][INFO]
[[4.   2.   1.  ]
 [0.   2.5  3.75]
 [0.   0.   2.5 ]]
Matriz L:
[[1.   0.   0.  ]
 [0.25 1.   0.  ]
 [0.5  0.   1.  ]]

Matriz U:
[[4.   2.   1.  ]
 [0.   2.5  3.75]
 [0.   0.   2.5 ]]
[01-13 21:44:01][INFO] Sustitución hacia adelante
[01-13 21:44:01][INFO] y =
[[ 7.  ]
 [-2.75]
 [ 4.5 ]]
[01-13 21:44:01][INFO] Sustitución hacia atrás
[01-13 21:44:01][INFO] i = 1
[01-13 21:44:01][INFO] suma = [6.75]
[01-13 21:44:01][INFO] U[i, i] = 2.5
[01-13 21:44:01][INFO] y[i] = [-2.75]
[01-13 21:44:01][INFO] i = 0
[01-13 21:44:01][INFO] suma = [-5.8]
[01-13 21:44:01][INFO] U[i, i] = 4.0
[01-13 21:44:01][INFO] y[i] = [7.]
```

```
Solución del sistema (vector X):
[[ 3.2]
 [-3.8]
 [ 1.8]]
[01-13 21:44:01][INFO]
[[4.   2.   1.  ]
 [0.   2.5  3.75]
 [0.   0.   2.5 ]]
[01-13 21:44:01][INFO]
[[4.   2.   1.  ]
 [0.   2.5  3.75]
 [0.   0.   2.5 ]]
Matriz L:
[[1.   0.   0.  ]
 [0.25 1.   0.  ]
 [0.5  0.   1.  ]]

Matriz U:
[[4.   2.   1.  ]
 [0.   2.5  3.75]
 [0.   0.   2.5 ]]
[01-13 21:44:01][INFO] Sustitución hacia adelante
[01-13 21:44:01][INFO] y =
[[ 7.  ]
 [-2.75]
 [ 4.5 ]]
[01-13 21:44:01][INFO] Sustitución hacia atrás
[01-13 21:44:01][INFO] i = 1
[01-13 21:44:01][INFO] suma = [6.75]
[01-13 21:44:01][INFO] U[i, i] = 2.5
[01-13 21:44:01][INFO] y[i] = [-2.75]
[01-13 21:44:01][INFO] i = 0
[01-13 21:44:01][INFO] suma = [-5.8]
[01-13 21:44:01][INFO] U[i, i] = 4.0
[01-13 21:44:01][INFO] y[i] = [7.]

Solución del sistema (vector X):
[[ 3.2]
 [-3.8]
 [ 1.8]]
```

B4)

[38]:

```
from src.linear_syst_methods import descomposicion_LU, resolver_LU,␣
 ↪matriz_aumentada
import numpy as np

A4 = [[2, 4, 6, 1], [4, 7, 5, -6], [2, 5, 18, 10], [6, 12, 38, 16]]
```

```python
b1 = [1, 2, 4, 5]

Ab4 = matriz_aumentada(A4, b1)

L4, U4 = descomposicion_LU(A4)

print("Matriz L:")
print(L4)

print("\nMatriz U:")
print(U4)

solucion4 = resolver_LU(L4, U4, b1)

print("\nSolución del sistema (vector X):")
print(solucion4)
```

```
[01-13 21:30:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  1. 12.  9.]
 [ 0.  0. 20. 13.]]
[01-13 21:30:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0. 20. 13.]]
[01-13 21:30:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 21:30:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
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.]]
```

```
[01-13 21:30:32][INFO] Sustitución hacia adelante
[01-13 21:30:32][INFO] y =
[[  1.]
 [  0.]
 [  3.]
 [-10.]]
[01-13 21:30:32][INFO] Sustitución hacia atrás
[01-13 21:30:32][INFO] i = 2
[01-13 21:30:32][INFO] suma = [-1.11111111]
[01-13 21:30:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0. 20. 13.]]
[01-13 21:30:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 21:30:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
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.]]
[01-13 21:30:32][INFO] Sustitución hacia adelante
[01-13 21:30:32][INFO] y =
[[  1.]
 [  0.]
 [  3.]
 [-10.]]
[01-13 21:30:32][INFO] Sustitución hacia atrás
[01-13 21:30:32][INFO] i = 2
[01-13 21:30:32][INFO] suma = [-1.11111111]
[01-13 21:30:32][INFO] U[i, i] = 5.0
[01-13 21:30:32][INFO] y[i] = [3.]
[01-13 21:30:32][INFO] i = 1
[01-13 21:30:32][INFO] suma = [3.13333333]
```

```
[01-13 21:30:32][INFO] U[i, i] = -1.0
[01-13 21:30:32][INFO] y[i] = [0.]
[01-13 21:30:32][INFO] i = 0
[01-13 21:30:32][INFO] suma = [16.35555556]
[01-13 21:30:32][INFO] U[i, i] = 2.0
[01-13 21:30:32][INFO] y[i] = [1.]

Solución del sistema (vector X):
[[-7.67777778]
 [ 3.13333333]
 [ 0.82222222]
 [-1.11111111]]
```

B4)

```
[39]: from src.linear_syst_methods import descomposicion_LU, resolver_LU,␣
       ↪matriz_aumentada
      import numpy as np

      A4 = [[2, 4, 6, 1], [4, 7, 5, -6], [2, 5, 18, 10], [6, 12, 38, 16]]
      b2 = [3, -5, 2, 6]

      Ab4 = matriz_aumentada(A4, b2)

      L4, U4 = descomposicion_LU(A4)

      print("Matriz L:")
      print(L4)

      print("\nMatriz U:")
      print(U4)

      solucion4 = resolver_LU(L4, U4, b2)

      print("\nSolución del sistema (vector X):")
      print(solucion4)
```

```
[01-13 21:31:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  1. 12.  9.]
 [ 0.  0. 20. 13.]]
[01-13 21:31:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0. 20. 13.]]
[01-13 21:31:32][INFO]
```

```
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 21:31:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
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.]]
[01-13 21:31:32][INFO] Sustitución hacia adelante
[01-13 21:31:32][INFO] y =
[[  3.]
 [-11.]
 [-12.]
 [ 45.]]
[01-13 21:31:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0. 20. 13.]]
[01-13 21:31:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 21:31:32][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
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.]]
[01-13 21:31:32][INFO] Sustitución hacia adelante
[01-13 21:31:32][INFO] y =
[[  3.]
 [-11.]
 [-12.]
 [ 45.]]
[01-13 21:31:32][INFO] Sustitución hacia atrás
[01-13 21:31:32][INFO] i = 2
[01-13 21:31:32][INFO] suma = [5.]
[01-13 21:31:32][INFO] U[i, i] = 5.0
[01-13 21:31:32][INFO] y[i] = [-12.]
[01-13 21:31:32][INFO] i = 1
[01-13 21:31:32][INFO] suma = [-16.2]
[01-13 21:31:32][INFO] U[i, i] = -1.0
[01-13 21:31:32][INFO] y[i] = [-11.]
[01-13 21:31:32][INFO] i = 0
[01-13 21:31:32][INFO] suma = [-36.2]
[01-13 21:31:32][INFO] U[i, i] = 2.0
[01-13 21:31:32][INFO] y[i] = [3.]

Solución del sistema (vector X):
[[19.6]
 [-5.2]
 [-3.4]
 [ 5. ]]
```

B4)

[40]:

```
from src.linear_syst_methods import descomposicion_LU, resolver_LU,␣
 ↪matriz_aumentada
import numpy as np

A4 = [[2, 4, 6, 1], [4, 7, 5, -6], [2, 5, 18, 10], [6, 12, 38, 16]]
b3 = [7, 8, -1, 0]

Ab4 = matriz_aumentada(A4, b3)

L4, U4 = descomposicion_LU(A4)

print("Matriz L:")
print(L4)

print("\nMatriz U:")
print(U4)
```

```python
solucion4 = resolver_LU(L4, U4, b3)

print("\nSolución del sistema (vector X):")
print(solucion4)
```

```
[01-13 21:32:40][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  1. 12.  9.]
 [ 0.  0. 20. 13.]]
[01-13 21:32:40][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0. 20. 13.]]
[01-13 21:32:40][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 21:32:40][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
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.]]
[01-13 21:32:40][INFO] Sustitución hacia adelante
[01-13 21:32:40][INFO] y =
[[  7.]
 [ -6.]
 [-14.]
 [ 35.]]
[01-13 21:32:40][INFO] Sustitución hacia atrás
[01-13 21:32:40][INFO] i = 2
[01-13 21:32:40][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0. 20. 13.]]
[01-13 21:32:40][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
[01-13 21:32:40][INFO]
[[ 2.  4.  6.  1.]
 [ 0. -1. -7. -8.]
 [ 0.  0.  5.  1.]
 [ 0.  0.  0.  9.]]
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.]]
[01-13 21:32:40][INFO] Sustitución hacia adelante
[01-13 21:32:40][INFO] y =
[[  7.]
 [ -6.]
 [-14.]
 [ 35.]]
[01-13 21:32:40][INFO] Sustitución hacia atrás
[01-13 21:32:40][INFO] i = 2
[01-13 21:32:40][INFO] suma = [3.88888889]
[01-13 21:32:40][INFO] U[i, i] = 5.0
[01-13 21:32:40][INFO] y[i] = [-14.]
[01-13 21:32:40][INFO] i = 1
[01-13 21:32:40][INFO] suma = [-6.06666667]
[01-13 21:32:40][INFO] U[i, i] = -1.0
[01-13 21:32:40][INFO] y[i] = [-6.]
[01-13 21:32:40][INFO] i = 0
[01-13 21:32:40][INFO] suma = [-17.84444444]
[01-13 21:32:40][INFO] U[i, i] = 2.0
[01-13 21:32:40][INFO] y[i] = [7.]

Solución del sistema (vector X):
[[12.42222222]
 [-0.06666667]
 [-3.57777778]
 [ 3.88888889]]
```