Taller 6

Moisés Pineda

Tabla de Contenidos

Escuela Politécnica Nacional

Nombre: Moisés Pineda

Fecha: 24/06/2025

Repositorio: https://github.com/SantiagoTmg/Metodos_Numericos_GRCC1/tree/main/Talleres/Taller6

¿Qué sucede al usar el código provisto de la eliminación gaussiana para el siguiente sistema de ecuaciones

$$4x_1 + x_2 + 2x_3 = 9$$
$$2x_1 + 4x_2 - x_3 = -5$$
$$x_1 + x_2 - 3x_3 = -9\ ?$$

Figura 1: image.png

```
%load_ext autoreload
```

```
%autoreload 2
from src import eliminacion_gaussiana

Ab = [[4, 1, 2, 9], [2, 4, -1, -5], [1, 1, -3, -9.0]]

eliminacion_gaussiana(Ab)
```

```
[06-24 17:34:32][INFO] 2025-06-24 17:34:32.477655
[06-24 17:34:32][INFO]
[[ 1.  1. -3. -9.]
 [ 0.  2.  5. 13.]
 [ 0. -3. 14. 45.]]
[06-24 17:34:32][INFO]
[[ 1.   1.  -3.  -9. ]
 [ 0.   2.   5.  13. ]
 [ 0.   0.  21.5 64.5]]

array([ 1., -1.,  3.])
```

Respuesta: Se llega a una solución que tiene un alto grado de error

$$
\begin{aligned}
2x_1 + x_2 - x_3 + x_4 - 3x_5 &= 7,\\
x_1 + 2x_3 - x_4 + x_5 &= 2,\\
-2x_2 - x_3 + x_4 - x_5 &= -5,\\
3x_1 + x_2 - 4x_3 + 5x_5 &= 6,\\
x_1 - x_2 - x_3 - x_4 + x_5 &= 3.
\end{aligned}
$$

Figura 2: image.png

```
%autoreload 3
from src import eliminacion_gaussiana

Ab = [[2.0, 1, -1, 1, -3, 7], [1, 0, 2, -1, 1, 2], [0, -2, -1, 1, -1, -5], [3, 1, -4, 0, 5, 6

solucion, intercambios = eliminacion_gaussiana(Ab)
```

```
print("Solución:", solucion)
print("Intercambios:", intercambios)
```

```
[06-24 17:15:54][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.   1.]]
[06-24 17:15:54][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.   4.]]
[06-24 17:15:54][INFO]
[[  1.    0.    2.   -1.    1.    2. ]
 [  0.    1.   -5.    3.   -5.    3. ]
 [  0.    0.   -5.    0.    7.   -3. ]
 [  0.    0.    0.    7.  -26.4   7.6]
 [  0.    0.    0.    3.  -16.2   8.8]]
[06-24 17:15:54][INFO]
[[  1.           0.           2.          -1.           1.
    2.        ]
 [  0.           1.          -5.           3.          -5.
    3.        ]
 [  0.           0.          -5.           0.           7.
   -3.        ]
 [  0.           0.           0.           3.         -16.2
    8.8       ]
 [  0.           0.           0.           0.          11.4
  -12.93333333]]
Solución: [ 1.91812865  1.96491228 -0.98830409 -3.19298246 -1.13450292]
Intercambios: [(0, 1), (2, 3), (3, 4)]
```

```
%autoreload 1
from src import eliminacion_gaussiana

Ab = [[2.0, 1, -1, 1, -3, 7], [1, 0, 2, -1, 1, 2], [0, -2, -1, 1, -1, -5], [3, 1, -4, 0, 5, 6

solucion, operaciones = eliminacion_gaussiana(Ab)
```

```
print("Solución:", solucion)
print("Operaciones:", operaciones)
```

```
[06-24 17:25:13][INFO] 2025-06-24 17:25:13.181089
[06-24 17:25:13][INFO]
[[  1.   0.   2.  -1.   1.   2.]
 [  2.   1.  -5.   3.  -5.   3.]
 [  0.  -2.  -1.   1.  -1.  -5.]
 [  0.   1. -10.   3.   2.   0.]
 [  0.  -1.  -3.   0.   0.   1.]]
[06-24 17:25:13][INFO]
[[  1.   0.   2.  -1.   1.   2.]
 [  2.   1.  -5.   3.  -5.   3.]
 [  0.   0. -11.   7. -11.   1.]
 [  0.   0.  -5.   0.   7.  -3.]
 [  0.   0.  -8.   3.  -5.   4.]]
[06-24 17:25:13][INFO]
[[  1.    0.    2.   -1.    1.    2. ]
 [  2.    1.   -5.    3.   -5.    3. ]
 [  0.    0.   -5.    0.    7.   -3. ]
 [  0.    0.    0.    7.  -26.4   7.6]
 [  0.    0.    0.    3.  -16.2   8.8]]
[06-24 17:25:13][INFO]
[[  1.           0.           2.          -1.           1.
    2.        ]
 [  2.           1.          -5.           3.          -5.
    3.        ]
 [  0.           0.          -5.           0.           7.
   -3.        ]
 [  0.           0.           0.           3.         -16.2
    8.8       ]
 [  0.           0.           0.           0.          11.4
  -12.93333333]]
Solución: [ 1.91812865  1.96491228 -0.98830409 -3.19298246 -1.13450292]
Operaciones: {'mult_div': 65, 'suma_resta': 50}
```

Comparación

```
import numpy as np
import time
import matplotlib.pyplot as plt
```

```python
# Configuración del experimento
np.random.seed(42)  # Para reproducibilidad
min_var = 2
max_var = 100
step = 5
sizes = range(min_var, max_var + 1, step)
tiempos = []

for n in sizes:
    # Generar matriz aleatoria bien condicionada
    A = np.random.rand(n, n + 1) * 10 - 5  # Valores entre -5 y 5

    # Medir tiempo
    start_time = time.time()
    try:
        sol = eliminacion_gaussiana(A.copy())
        elapsed = time.time() - start_time
        tiempos.append(elapsed)
        print(f"n = {n}: {elapsed:.6f} segundos")
    except ValueError as e:
        print(f"Error con n={n}: {str(e)}")
        tiempos.append(np.nan)

# Filtrar valores no válidos
valid_sizes = [s for s, t in zip(sizes, tiempos) if not np.isnan(t)]
valid_times = [t for t in tiempos if not np.isnan(t)]

# Gráfico de tiempo vs tamaño
plt.figure(figsize=(12, 6))
plt.plot(valid_sizes, valid_times, 'bo-')
plt.xlabel('Número de variables (n)')
plt.ylabel('Tiempo de ejecución (segundos)')
plt.title('Rendimiento de Eliminación Gaussiana')
plt.grid(True)
```

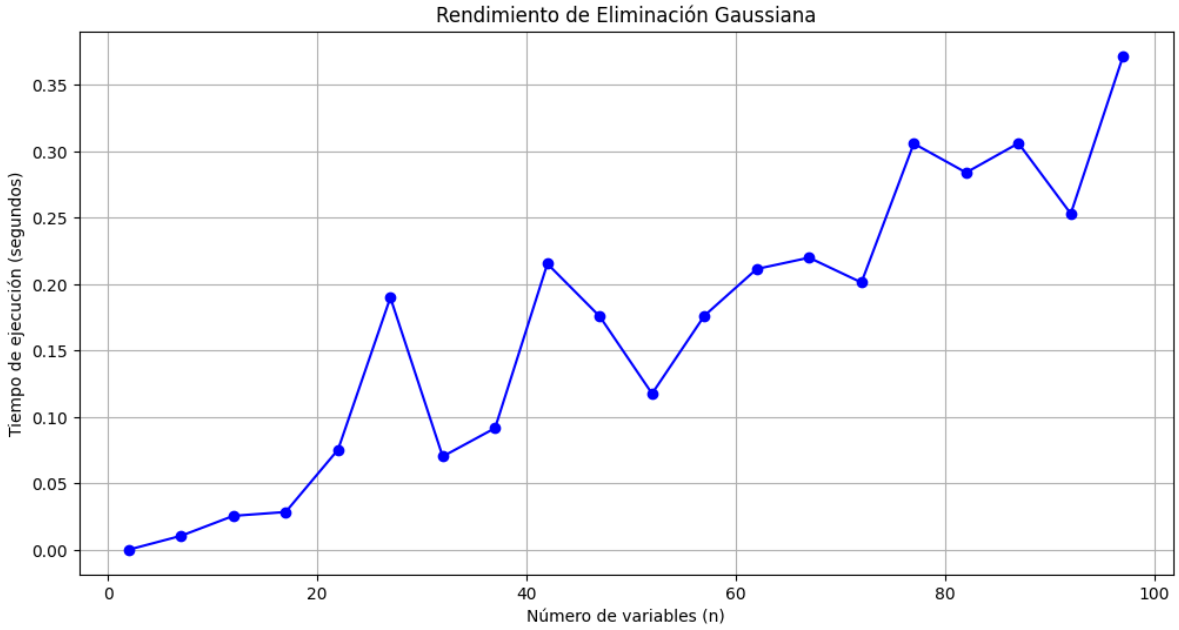

Figura 3: image.png

```
%autoreload 1
from src import gauss_jordan

# Configuración del experimento
np.random.seed(42)  # Para reproducibilidad
min_var = 2
max_var = 100
step = 5
sizes = range(min_var, max_var + 1, step)
tiempos = []

for n in sizes:
    # Generar matriz aleatoria bien condicionada
    Ab = np.random.rand(n, n + 1) * 10 - 5  # Valores entre -5 y 5

    # Medir tiempo
    start_time = time.time()
    try:
        sol = gauss_jordan(Ab.copy())
        elapsed = time.time() - start_time
        tiempos.append(elapsed)
```

```
        print(f"n = {n}: {elapsed:.6f} segundos")
    except ValueError as e:
        print(f"Error con n={n}: {str(e)}")
        tiempos.append(np.nan)

# Filtrar valores no válidos
valid_sizes = [s for s, t in zip(sizes, tiempos) if not np.isnan(t)]
valid_times = [t for t in tiempos if not np.isnan(t)]

# Gráfico de tiempo vs tamaño
plt.figure(figsize=(12, 6))
plt.plot(valid_sizes, valid_times, 'go-', label='Gauss-Jordan')
plt.xlabel('Número de variables (n)')
plt.ylabel('Tiempo de ejecución (segundos)')
plt.title('Rendimiento de Gauss-Jordan vs Tamaño del Sistema')
plt.grid(True)
```

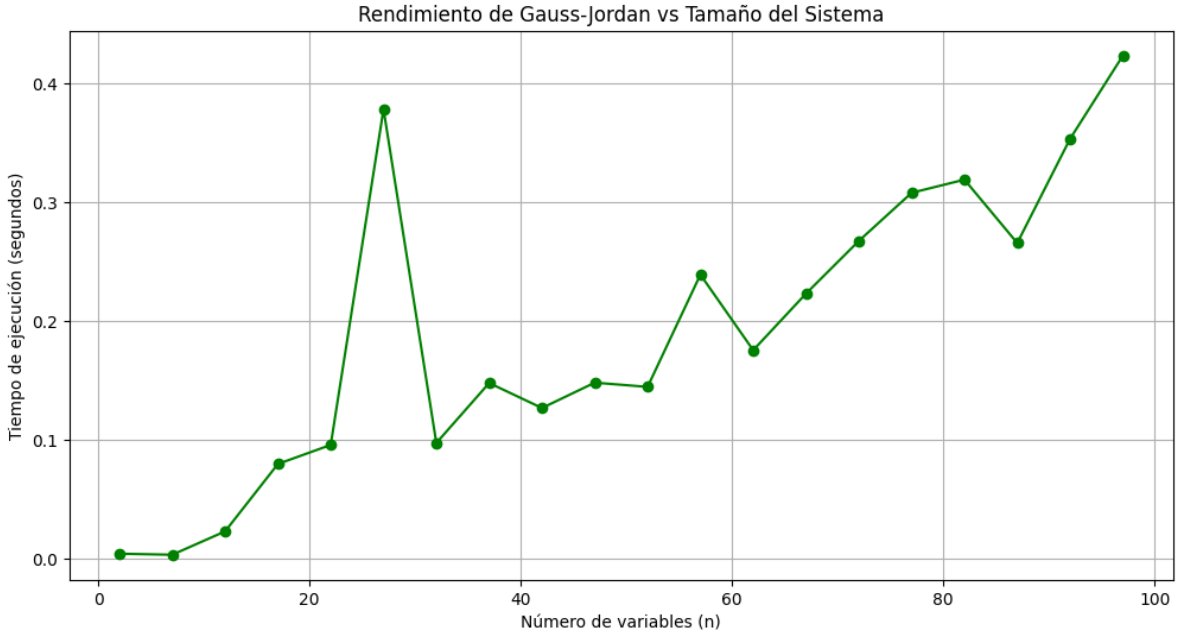

Figura 4: image.png