MANUAL DE USUARIO

Sistema de Resolución de Ecuaciones Lineales

Métodos: Gauss, Gauss-Jordan y LU

ÍNDICE

---

1. INTRODUCCIÓN

Este programa permite resolver sistemas de ecuaciones lineales de la forma **Ax = b** utilizando tres métodos numéricos clásicos:

- **Eliminación de Gauss**: Método directo basado en eliminación hacia adelante y sustitución hacia atrás
- **Gauss-Jordan**: Método que reduce la matriz a su forma escalonada reducida
- **Descomposición LU**: Factoriza la matriz A en el producto de dos matrices triangulares

¿Para qué sirve?

Resolver sistemas de ecuaciones lineales es fundamental en:

- Ingeniería (análisis estructural, circuitos eléctricos)
- Economía (modelos de equilibrio)
- Física (sistemas dinámicos)
- Ciencias de datos (regresión lineal)

---

2. REQUISITOS DEL SISTEMA

Software Necesario:

- **Python 3.7 o superior**
- **NumPy** (biblioteca de cálculo numérico)

Instalación de Dependencias:

```
pip install numpy
```

Verificar Instalación:

```
python --version
python -c "import numpy; print('NumPy version:', numpy.__version__)"
```

---

3. INSTALACIÓN Y EJECUCIÓN

Estructura de Archivos:

```
SistemasEcuaciones_1/
├── main.py
└── metodos/
    ├── gauss.py
    ├── gauss_jordan.py
    └── lu.py
```

Ejecución del Programa:

Opción 1: Desde la terminal

```
cd SistemasEcuaciones_1
python main.py
```

Opción 2: Desde un IDE

- Abrir el archivo `main.py` en PyCharm, VSCode o Spyder
- Ejecutar el archivo (botón Run o F5)

---

4. DESCRIPCIÓN DE LOS MÉTODOS

4.1 Método de Eliminación de Gauss

Funcionamiento:

1. Convierte la matriz A en una matriz triangular superior
2. Utiliza sustitución hacia atrás para encontrar las soluciones

Ventajas:

- Eficiente para sistemas medianos

- Fácil de implementar
- Complejidad $O(n^3)$

Limitaciones:

- Puede tener problemas con pivotes pequeños
- No calcula explícitamente la matriz inversa

4.2 Método de Gauss-Jordan

Funcionamiento:

1. Reduce la matriz aumentada [A|b] a su forma escalonada reducida
2. La solución se lee directamente de la última columna

Ventajas:

- Proporciona la solución directamente
- Útil para calcular la matriz inversa
- No requiere sustitución hacia atrás

Limitaciones:

- Más operaciones que Gauss simple
- Complejidad $O(n^3)$

4.3 Descomposición LU

Funcionamiento:

1. Factoriza $A = L \times U$ (triangular inferior × triangular superior)
2. Resuelve $Ly = b$ (sustitución hacia adelante)
3. Resuelve $Ux = y$ (sustitución hacia atrás)

Ventajas:

- Muy eficiente para resolver múltiples sistemas con la misma matriz A
- La descomposición se calcula una sola vez
- Útil en métodos iterativos

Limitaciones:

- Requiere más memoria que Gauss
- No todas las matrices tienen descomposición LU sin pivoteo

---

5. GUÍA DE USO PASO A PASO

Paso 1: Iniciar el Programa

Al ejecutar `main.py`, verá el menú principal:

```
--- SISTEMAS DE ECUACIONES LINEALES ---
1. Método de Eliminación de Gauss
2. Método de Gauss-Jordan
3. Descomposición LU
4. Salir
Seleccione una opción:
```

Paso 2: Seleccionar el Método

Ingrese el número correspondiente al método deseado (1, 2 o 3).

Paso 3: Ingresar el Tamaño del Sistema

```
Ingrese el número de ecuaciones: 3
```

Esto define un sistema de 3 ecuaciones con 3 incógnitas.

Paso 4: Ingresar los Coeficientes

El programa solicitará los coeficientes de la matriz A y el vector b:

```
Ingrese los coeficientes del sistema Ax = b:
A[1,1]: 2
A[1,2]: 1
A[1,3]: -1
b[1]: 8
A[2,1]: -3
A[2,2]: -1
A[2,3]: 2
b[2]: -11
...
```

Importante: Ingrese los valores con cuidado, ya que un error afectará el resultado.

Paso 5: Visualizar la Solución

El programa mostrará la solución del sistema:

```
Solución del sistema:
[ 2.  3. -1.]
```

Esto significa: $x_1=2$, $x_2=3$, $x_3=-1$

---

6. INTERPRETACIÓN DE RESULTADOS

Formato de Salida

La solución se presenta como un vector NumPy:

$[x_1\ x_2\ x_3\ \dots\ x_n]$

Verificación Manual

Para verificar la solución, sustituya los valores en las ecuaciones originales:

Ejemplo: Si el sistema es:

2x + y - z = 8

Y la solución es x=2, y=3, z=-1:

2(2) + 3 - (-1) = 4 + 3 + 1 = 8 ✓

Casos Especiales

Sistema sin solución: El programa puede mostrar un error o valores inconsistentes.

Sistema con infinitas soluciones: Los métodos pueden no converger o dar resultados inesperados.

Matriz singular: Si el determinante es cero, el sistema no tiene solución única.

---

7. SOLUCIÓN DE PROBLEMAS

Problema 1: Error "ZeroDivisionError"

Causa: Pivote cero en la diagonal de la matriz. **Solución:**

- Reordene las ecuaciones
- Use un método con pivoteo parcial
- Verifique que el sistema tenga solución única

Problema 2: "ModuleNotFoundError: No module named 'numpy'"

Causa: NumPy no está instalado. **Solución:**

```
pip install numpy
```

Problema 3: Resultados Incorrectos

Causa: Error en el ingreso de datos. **Solución:**

- Verifique cada coeficiente cuidadosamente
- Use decimales con punto (.) no coma (,)
- Ejemplo correcto: 3.5, no 3,5

Problema 4: Números Muy Grandes o Muy Pequeños

Causa: Mal condicionamiento de la matriz. **Solución:**

- Escale las ecuaciones (multiplique por constantes)
- Use precisión extendida si es necesario

---

8. EJEMPLOS PRÁCTICOS

Ejemplo Completo: Sistema 2x2

Sistema a resolver:

```
3x + 2y = 18
x + 4y = 16
```

Ejecución:

1. Ejecutar: `python main.py`
2. Seleccionar: `1` (Gauss)
3. Ingresar: `2` (número de ecuaciones)
4. Ingresar coeficientes:
5. A[1,1]: 3A[1,2]: 2b[1]: 18A[2,1]: 1A[2,2]: 4b[2]: 16
6. Resultado:
7. Solución del sistema:[4. 3.]

   Interpretación: x = 4, y = 3

Verificación:

```
3(4) + 2(3) = 12 + 6 = 18
4 + 4(3) = 4 + 12 = 16
```

SOPORTE Y CONTACTO

Para más información sobre métodos numéricos:

- Documentación de NumPy: https://numpy.org/doc/
- Álgebra Lineal Numérica: Golub & Van Loan

NOTAS FINALES

- Guarde copias de sus sistemas de ecuaciones antes de resolverlos
- Para sistemas grandes (n > 100), considere métodos iterativos
- Los resultados son aproximaciones numéricas con precisión de máquina
- Siempre verifique la solución sustituyendo en las ecuaciones originales

---

Versión del Manual: 1.0
Fecha: Octubre 2025
Compatible con: Python 3.7+, NumPy 1.19+