Sistemas de punto flotante y sistemas ecuaciones lineales

Isabela Acevedo García, Juan Manuel Cuellar

Pontificia Universidad Javeriana Cali

Cali, Colombia

isa43461@javerianacali.edu.co
juanma1909@javerianacali.edu.co

***Abstract*—** En el presente documento se explicará los procesos que se llevaron a cabo para resolver el problema de generar sistemas de números de punto flotante y el problema de resolver sistemas de ecuaciones lineales.

I. INTRODUCCIÓN

Se desarrollaron dos archivos en Python de tal forma que cada uno cumpliera con los requisitos solicitados, el primer archivo resuelve el problema de los sistemas de punto flotante, dando el sistema, la cantidad de números de punto flotante, el UFL y OFL. El segundo archivo fue acerca de los sistemas de ecuaciones lineales en el cual se desarrolló la singularidad de una matriz, la permutación, matrices triangulares, Gauss y Gauss Jordan.

II. DESARROLLO DE CONTENIDO.

A. Materiales y métodos.

➔ **Materiales:**

- ***Python:*** Este fue el lenguaje utilizado para desarrollar los algoritmos de sistemas de punto flotante y de ecuaciones lineales. *"Python es un lenguaje de programación de código abierto, orientado a objetos, muy simple y fácil de entender. Tiene una sintaxis sencilla que cuenta con una vasta biblioteca de herramientas, que hacen de Python un lenguaje de programación único."* [1].
- ***Función round***: Esta es una función de Python utilizada para redondear los dígitos. *"La función **round** redondea un número a una precisión dada por el número de cifras decimales indicadas como argumento. Si no se indica el número de cifras decimales, se toma 0 como valor por defecto."* [2].
- ***Calculadora de eliminación de Gauss-Jordan:*** Esta fue utilizada para verificar que con los datos de entrada los resultados de la función fueran correctos. *"Puedes comprobar tu sistema lineal de ecuaciones en consistencia usando nuestra Calculadora de eliminación de Gauss-Jordan."*[3].
- ***Librería Numpy:*** Esta librería nos ayudó a hacer operaciones de álgebra lineal de forma eficiente. *"NumPy es un paquete de Python que significa "Numerical Python", es la librería principal para la informática científica, proporciona potentes estructuras de datos, implementando matrices y matrices multidimensionales. Estas estructuras de datos garantizan cálculos eficientes con matrices." [4].*
- ***Librería Matplotlib:*** Esta librería nos ayudó a graficar el sistema numérico listado. *"Matplotlib es una librería de trazado utilizada para gráficos 2D en lenguaje de programación Python, es muy flexible y tiene muchos valores predeterminados incorporados que te ayudarán muchísimo en tu trabajo."* [5].

➔ **Metodología**

Para la elaboración del algoritmo generador de sistemas de punto flotante, se hizo uso de la fórmula vista durante la clase de Computación científica, dicha fórmula depende de 4 parámetros principales: β, t, L y U, siendo L y U los responsables del rango. la fórmula nombrada es la siguiente:

$$X = \pm (d0 + \frac{d1}{\beta} + \frac{d2}{\beta^2} + ... + \frac{dt-1}{\beta^{t-1}})\beta^e$$

donde *e* es un número, el cual va aumentando dentro del rango del sistema. Para la implementación se realizó un ciclo desde U hasta L, durante cada iteración del ciclo generaban todos los posibles números binarios que se pudieran generar con t-1 posiciones binarias, a partir de dichos números se calculaba la mantisa, y con ello ya se calculaban los distintos x.

Con respecto a la elaboración de los algoritmos que solucionan sistemas de ecuaciones lineales, se implementaron 2 algoritmos distintos, el de Gauss y el de Gauss-Jordan.

En primer lugar, el de Gauss se basa en la generación de una matriz triangular, ya sea esta superior o inferior, a partir de una matriz a, dicha matriz triangular posteriormente se resuelve usando los métodos de sustitución sucesiva. Para llevar a cabo esto se hacen métodos de aniquilación de columnas a través de la multiplicación de matrices de eliminación, esto siempre tomando en cuenta las situaciones cuando sea necesario permutar las filas de la matriz, esto cuando el pivote es 0.

El de Gauss-Jordan, por otro lado, se basa en la generación de una matriz diagonal, a partir de una matriz a, la solución de dicha matriz es sencilla pues cada xi está asociado con un valor en el vector b, de igual forma que en Gauss, se utilizan métodos de aniquilación de columnas, también tomando en cuenta la permutación de filas, cuando esta es necesaria.

Para más información sobre la implementación en código se puede revisar la documentación que está explícita en el código mismo.

B. *Resultados de sistemas de punto flotante.*

➢ ***Ejemplo número 1***

- $\beta = 2$, T = 3, L = -1, U = 1
- N: 25
- UFL: 0.5
- OFL: 3.5
- Sistema numérico listado: [0, 0.5, -0.5, 0.625, -0.625, 0.75, -0.75, 0.875, -0.875, 1.0, -1.0, 1.25, -1.25, 1.5, -1.5, 1.75, -1.75, 2.0, -2.0, 2.5, -2.5, 3.0, -3.0, 3.5, -3.5]

Fig. 1 Es el ejemplo #1 graficado en Python con la librería matplotlib.

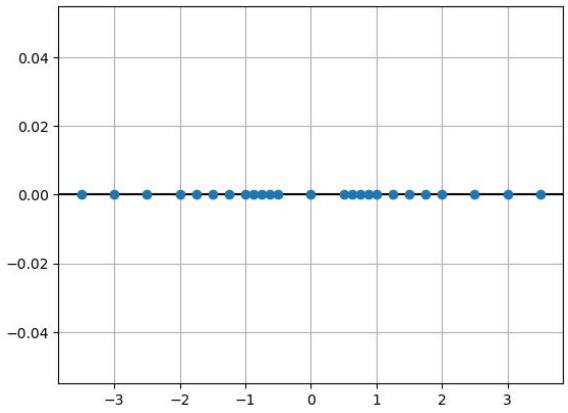

Fig. 1 Ejemplo visto en clase.

- ***Análisis y discusión***
  Este es un sistema bastante pequeño con una precisión respetable, esto debido a Rango y al parámetro t de la función.

➢ ***Ejemplo número 2***

- $\beta = 2$, T = 5, L = -3, U = 3
- n: 225
- UFL: 0.125
- OFL: 15.5
- Sistema numérico listado: [0, 0.125, -0.125, 0.1328125, -0.1328125, 0.140625, -0.140625, 0.1484375, -0.1484375, 0.15625, -0.15625, 0.1640625, -0.1640625, 0.171875, -0.171875, 0.1796875, -0.1796875, 0.1875, -0.1875, 0.1953125, -0.1953125, 0.203125, -0.203125, 0.2109375, -0.2109375, 0.21875, -0.21875, 0.2265625, -0.2265625, 0.234375, -0.234375, 0.2421875, -0.2421875, 0.25, -0.25, 0.265625, -0.265625, 0.28125, -0.28125, 0.296875, -0.296875, 0.3125, -0.3125, 0.328125, -0.328125, 0.34375, -0.34375, 0.359375, -0.359375, 0.375, -0.375, 0.390625, -0.390625, 0.40625, -0.40625, 0.421875, -0.421875, 0.4375, -0.4375, 0.453125, -0.453125, 0.46875,

-0.46875, 0.484375, -0.484375, 0.5, -0.5, 0.53125, -0.53125, 0.5625, -0.5625, 0.59375, -0.59375, 0.625, -0.625, 0.65625, -0.65625, 0.6875, -0.6875, 0.71875, -0.71875, 0.75, -0.75, 0.78125, -0.78125, 0.8125, -0.8125, 0.84375, -0.84375, 0.875, -0.875, 0.90625, -0.90625, 0.9375, -0.9375, 0.96875, -0.96875, 1.0, -1.0, 1.0625, -1.0625, 1.125, -1.125, 1.1875, -1.1875, 1.25, -1.25, 1.3125, -1.3125, 1.375, -1.375, 1.4375, -1.4375, 1.5, -1.5, 1.5625, -1.5625, 1.625, -1.625, 1.6875, -1.6875, 1.75, -1.75, 1.8125, -1.8125, 1.875, -1.875, 1.9375, -1.9375, 2.0, -2.0, 2.125, -2.125, 2.25, -2.25, 2.375, -2.375, 2.5, -2.5, 2.625, -2.625, 2.75, -2.75, 2.875, -2.875, 3.0, -3.0, 3.125, -3.125, 3.25, -3.25, 3.375, -3.375, 3.5, -3.5, 3.625, -3.625, 3.75, -3.75, 3.875, -3.875, 4.0, -4.0, 4.25, -4.25, 4.5, -4.5, 4.75, -4.75, 5.0, -5.0, 5.25, -5.25, 5.5, -5.5, 5.75, -5.75, 6.0, -6.0, 6.25, -6.25, 6.5, -6.5, 6.75, -6.75, 7.0, -7.0, 7.25, -7.25, 7.5, -7.5, 7.75, -7.75, 8.0, -8.0, 8.5, -8.5, 9.0, -9.0, 9.5, -9.5, 10.0, -10.0, 10.5, -10.5, 11.0, -11.0, 11.5, -11.5, 12.0, -12.0, 12.5, -12.5, 13.0, -13.0, 13.5, -13.5, 14.0, -14.0, 14.5, -14.5, 15.0, -15.0, 15.5, -15.5]

Fig. 2 Es el ejemplo #2 graficado en Python con la librería matplotlib.

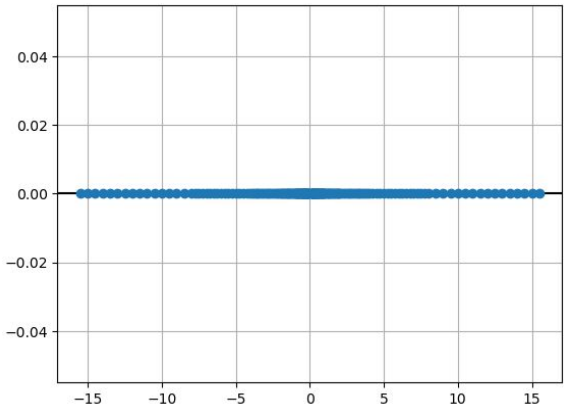

Fig. 2 Ejemplo.

- ***Análisis y discusión***
  En este sistema se puede ver cómo la utilización de un t mayor, hace que el número de cifras decimales por número sean mayores, dando así una gráfica como se muestra en la Figura 2, esto debido a la cercanía entre todos los puntos.

➢ ***Ejemplo número 3***

- β = 2, T = 5, L = 3, U = 3
- N: 33
- UFL: 8
- OFL: 15.5
- Sistema numérico listado: [0, 8.0, -8.0, 8.5, -8.5, 9.0, -9.0, 9.5, -9.5, 10.0, -10.0, 10.5, -10.5, 11.0, -11.0, 11.5, -11.5, 12.0, -12.0, 12.5, -12.5, 13.0, -13.0, 13.5, -13.5, 14.0, -14.0, 14.5, -14.5, 15.0, -15.0, 15.5, -15.5]

Fig. 3 Es el ejemplo #3 graficado en Python con la librería matplotlib.

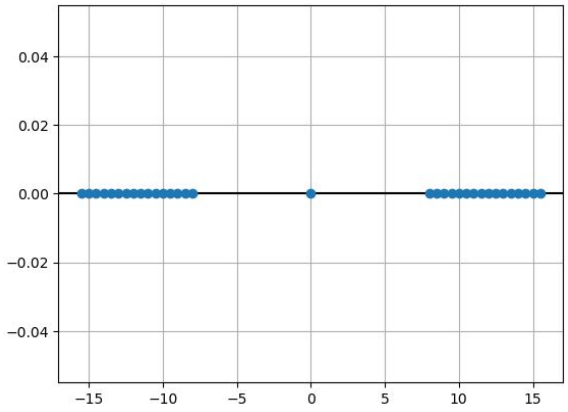

Fig. 3 Ejemplo.

- ***Análisis y discusión***
  Se puede ver que siempre que se utiliza un L positivo, el UFL es sustancialmente mayor, esto debido al hecho de que al final se multiplica la mantisa por β^e, como "e" empieza positivo y el resultado de la mantisa necesariamente es mayor que 1, significa que el UFL no podría ser menor a 1.

C. *Resultados de ecuaciones lineales*

➢ ***Ejemplo número 1***

- a = [[2,4,-2],[4,9,-3],[-2,-3,7]]
- b = [2,8,10]
- No es una matriz singular
- Gauss:
  - x1: -1.0
  - x2: 2.0
  - x3: 2.0
- Gauss-Jordan:
  - x1: -1.0

- x2: 2.0
- x3: 2.0

➢ ***Ejemplo número 2***

- a = [[5,4,-1,2,-9],[4,3,-5,-5,4],[6,-6,-3,-9,4],[-8,8,5,-3,-6],[-3,-7,2,-5,4]]
- b = [29,-65,-113,35,-50]
- No es una matriz singular
- Gauss:
  - x1: -2.0
  - x2: 2.0
  - x3: 3.0
  - x4: 8.0
  - x5: -2.0
- Gauss-Jordan:
  - x1: -2.0
  - x2: 2.0
  - x3: 3.0
  - x4: 8.0
  - x5: -2.0

➢ ***Ejemplo número 3.***

- a = [[0,9,-2],[0,4,1],[4,1,4]]
- b = [2,4,8]
- No es una matriz singular
- Gauss:
  - x1: 0.2059
  - x2: 0.5882
  - x3: 1.6471
- Gauss-Jordan:
  - x1: 0.2059
  - x2: 0.5882
  - x3: 1.6471

- ***Análisis y discusión***
  En esta matriz, se puede apreciar claramente el uso de la permutación de filas, pues desde el primer pivote a escoger toca realizar dicha permutación.

III. CONCLUSIONES

Finalmente, se puede ver de manera clara y concisa que los números de punto flotante dependen de una mantisa, un beta y un exponente. El número depende de 4 enteros, las cuales son: β que es la base o raíız, un T que vendría siendo la precisión. y [L,U] que sería el rango del exponente, con los enteros mencionados se puede hallar la cantidad de números flotantes, el UFL y OFL. También, vemos que con una matriz no singular se puede aplicar el método de Gauss y Gauss Jordan, o si en algún momento hay un pivote cero se debe hacer una permutación para arreglar esto y poder aplicar bien las funciones de Gauss y Gauss Jordan, también desarrollamos los sistemas de triángulos superior e inferior.

IV. REFERENCIAS

[1] A. Soloaga (2018, Octubre 19) “Principales Usos de Python”.[Online]. Disponible en: https://www.akademus.es/blog/programacion/principales-usos-python/

[2] D. Barrueco. (2019, Enero 19). “ROUND”. [Online]. Disponible en: https://www.interactivechaos.com/python/function/round

[3]. Matrix Reshish (2018, Mayo 18). “Calculadora de eliminación de Gauss-Jordan”[Online]. Disponible en: https://matrix.reshish.com/es/gauss-jordanElimination.php

[4]. L. González. (2018, Septiembre 21) “Introducción a la librería NumPy de Python – Parte 1 ``.[Online]. Disponible en: https://ligdigonzalez.com/introduccion-a-numpy-python-1/

[5]. L. González. (2018, Octubre 19). “Introducción a la Librería Matplotlib de Python – Parte 1 ``.[Online]. Disponible en: https://ligdigonzalez.com/libreria-pandas-de-matplotlib-tutorial/