LABORATORIO 1

Unidades 1 y 2

Nicolás Delgado
Camila Paladines

Computación Científica

Profesor: Hernán Darío Vargas Cardona, PhD

Marzo 12 de 2021

RESUMEN

En el presente informe se describen los resultados conseguidos al implementar algoritmos en Python para la obtención de respuestas a problemas matemáticos, como lo son el cómputo de los números de un Sistema de Punto Flotante y la resolución de Sistemas de Ecuaciones Lineales mediante diferentes métodos como Sustitución Sucesiva hacia atrás, Sustitución Sucesiva hacia adelante, Eliminación de Gauss y Eliminación de Gauss-Jordan. Además, se realizan algunos análisis sobre los métodos utilizados y la complejidad computacional que estos poseen, permitiendo comprender mejor el comportamiento de estos algoritmos dependiendo de sus datos de entrada.

ABSTRACT

This report describes the results achieved when implementing algorithms in Python to obtain answers to mathematical problems, such as the computation of the numbers of a Floating Point System and the resolution of Systems of Linear Equations using different methods such as Backward Successive Substitution, Forward Successive Substitution, Gaussian Elimination, and Gauss-Jordan Elimination. Also, some analysis is carried out on the methods used and the computational complexity they have, allowing a better understanding of the behavior of these algorithms according to their input data.

Contenido

1. Introducción

Durante mucho tiempo se utilizó el método clásico para resolver problemas matemáticos. En las últimas décadas, con la invención del computador se pudo automatizar y optimizar este proceso de manera considerable. Sin embargo, el "mundo" matemático, en su mayoría, es continuo, mientras que el "mundo" de los computadores es discreto, lo cual tiene un error inherente al momento de querer resolver problemas matemáticos, ya que los cálculos hechos por un computador no siempre son exactos.

En los inicios de la computación se contempló la necesidad de crear una rama especializada en cómo modelar toda la matemática en un computador, naciendo así la computación científica. Con ella, se originaron los sistemas de punto flotante, ya que se necesitó representar números reales dentro de las máquinas de alguna u otra manera. Además, teniendo la manera de representar estos números se han logrado resolver distintos problemas matemáticos, como los sistemas de ecuaciones lineales mediante diferentes métodos.

2. Unidad 1: Sistemas de Punto Flotante

2.1. Materiales y métodos

Materiales

Para el desarrollo de esta unidad se usó Python 3.7, con las librerías `matplotlib.pyplot` y `time`.

Métodos

La manera en que se representan los números en un computador es mediante un sistema de punto flotante, el cual se construye mediante cuatro enteros:

- β, la base o raíz
- t, la precisión de los números del sistema
- $[L, U]$ el rango del exponente

Dados estos valores, es posible construir un sistema de punto flotante, donde cada número se calcula mediante la siguiente ecuación:

$$x = \pm \left(d_0 + \frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \cdots + \frac{d_{t-1}}{\beta^{t-1}} \right) \beta^e$$

donde $0 \leq d_i \leq \beta - 1$, $i = 0, \ldots, t-1$, $L \leq e \leq U$.

Además, es posible calcular las propiedades de un sistema de punto flotante, como lo son:

- **N**: es la cantidad de números del sistema, se calcula mediante:

$$N = 2(\beta - 1)\beta^{t-1}(U - L + 1) + 1$$

- **UFL (Under Flow Level)**: es el número (positivo) más pequeño del sistema, se calcula mediante:

$$UFL = \beta^L$$

- **OFL (Over Flow Level)**: es el número (positivo) más grande del sistema, se calcula mediante:

$$OFL = \beta^{U+1}(1 - \beta^{-t})$$

También se usó una función auxiliar para calcular la secuencia binaria de un número dado para obtener los respectivos d_i de la ecuación mencionada anteriormente. Para este laboratorio se fijó un $\beta = 2$.

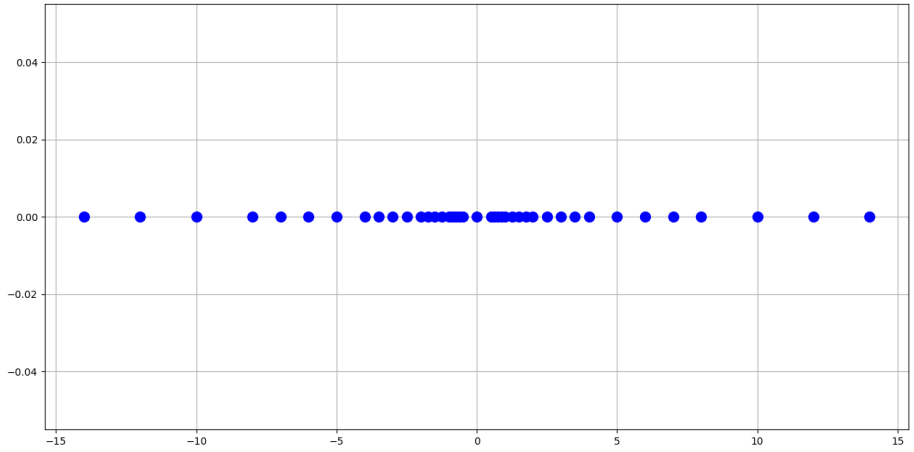
2.2. Resultados de las simulaciones

Ejemplo 1

Dados t = 3, L = -1 y U = 3 se obtienen los siguientes números del sistema de punto flotante:

-14	-12	-10	-8	-7	-6	-5	-4	-3.5	-3
-2.5	-2	-1.75	-1.5	-1.25	-1	-0.875	-0.75	-0.625	-0.5
0	0.5	0.625	0.75	0.875	1	1.25	1.5	1.75	2
2.5	3	3.5	4	5	6	7	8	10	12
14									

Que se pueden observar mejor en la siguiente gráfica:



Las propiedades del sistema son las siguientes:

- N = 41
- UFL = 0.5
- OFL = 14

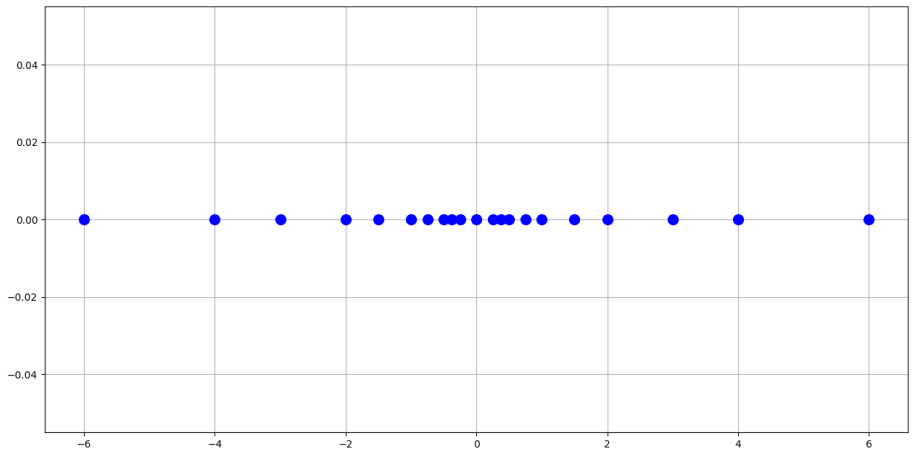
El algoritmo tarda 9.2 $\times 10^{-5}$s (aproximadamente) en realizar este cálculo.

Ejemplo 2

Dados t = 2, L = -2 y U = 2 se obtienen los siguientes números del sistema de punto flotante:

-6	-4	-3	-2	-1.5	-1	-0.75	-0.5	-0.375	-0.25
0	0.25	0.375	0.5	0.75	1	1.5	2	3	4
6									

Que se pueden observar mejor en la siguiente gráfica:



Las propiedades del sistema son las siguientes:

- N = 21
- UFL = 0.25
- OFL = 6

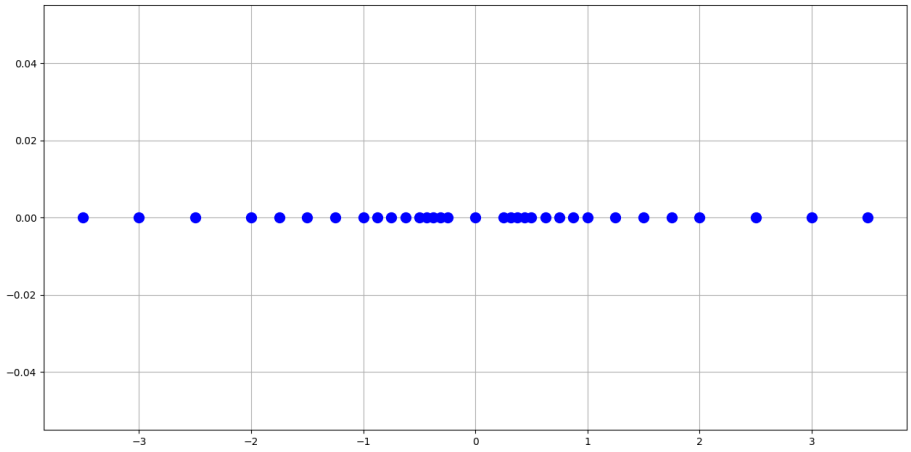
El algoritmo tarda 8.7 $\times 10^{-5}$s (aproximadamente) en realizar este cálculo.

Ejemplo 3

Dados t = 3, L = -2 y U = 1 se obtienen los siguientes números del sistema de punto flotante:

-3.5	-3	-2.5	-2	-1.75	-1.5	-1.25	-1	-0.875	-0.75
-0.625	-0.5	-0.4375	-0.375	-0.3125	-0.25	0	0.25	0.3125	0.375
0.4375	0.5	0.625	0.75	0.875	1	1.25	1.5	1.75	2
2.5	3	3.5							

Que se pueden observar mejor en la siguiente gráfica:



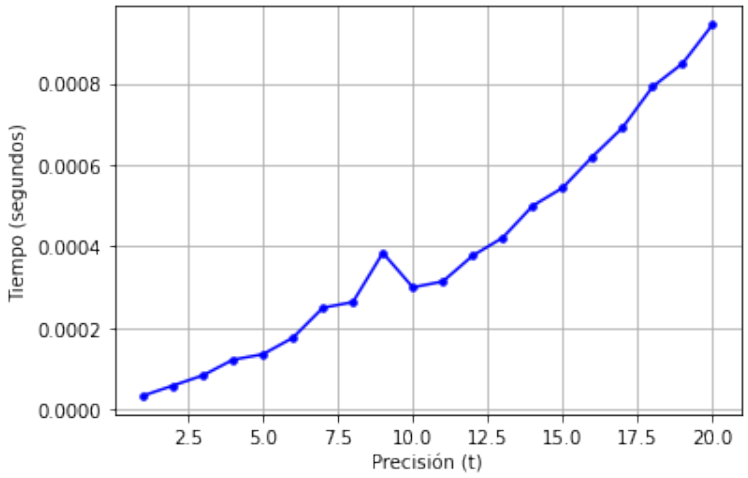
Las propiedades del sistema son las siguientes:

- N = 33
- UFL = 0.25
- OFL = 3.5

El algoritmo tarda 10.2 $\times 10^{-5}$s (aproximadamente) en realizar este cálculo.

2.3. Discusión y análisis

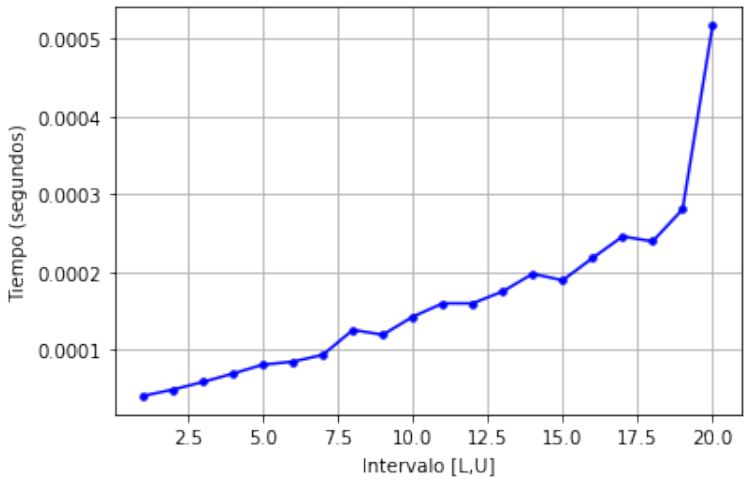
Se logró observar que la complejidad del algoritmo planteado es lineal (aproximadamente) respecto a la variable t, como se muestra en la siguiente gráfica:



Para t = 1. . . 20, L = -1 y U = 1 con los siguientes tiempos de ejecución:

Precisión t	Tiempo (s)
1	0.00003
2	0.00006
3	0.00008
4	0.00012
5	0.00014
6	0.00018
7	0.00025
8	0.00026
9	0.00038
10	0.00030
11	0.00031
12	0.00038
13	0.00042
14	0.00050
15	0.00054
16	0.00062
17	0.00069
18	0.00079
19	0.00085
20	0.00094

Además, la complejidad del algoritmo es también lineal (aproximadamente) respecto a las variables L y U si este intervalo es simétrico, como se muestra en la siguiente gráfica:



Para t = 1, [L,U]=[-1,1]... [-20, 20] con los siguientes tiempos de ejecución:

Rango $[L, U]$	Tiempo (s)
$[-1, 1]$	0.00004
$[-2, 2]$	0.00005
$[-3, 3]$	0.00006
$[-4, 4]$	0.00007
$[-5, 5]$	0.00008
$[-6, 6]$	0.00009
$[-7, 7]$	0.00009
$[-8, 8]$	0.00013
$[-9, 9]$	0.00012
$[-10, 10]$	0.00014
$[-11, 11]$	0.00016
$[-12, 12]$	0.00016
$[-13, 13]$	0.00018
$[-14, 14]$	0.00020
$[-15, 15]$	0.00019
$[-16, 16]$	0.00022
$[-17, 17]$	0.00025
$[-18, 18]$	0.00024
$[-19, 19]$	0.00028
$[-20, 20]$	0.00052

3. Unidad 2: Sistemas de Ecuaciones Lineales

3.1. Materiales y métodos

Materiales

Para el desarrollo de esta unidad se usó Python 3.7, con las librerías `numpy`, `time` y `matplotlib.pyplot`.

Métodos

3.1.1. Matriz Singular

Una matriz $\mathbf{A} \in \mathbb{R}^{m \times n}$ es singular si cumple alguna de las siguientes condiciones:

1. $\mathbf{A}$ no tiene inversa
2. $\det(\mathbf{A}) = 0$
3. Rango($\mathbf{A} < n$)

Si no cumple ninguna de las condiciones mencionadas, $\mathbf{A}$ es *no singular*, entonces su inversa A^{-1} existe, por lo que la solución a un sistema $Ax = b$ se obtiene mediante $x = A^{-1}b$.

3.1.2. Matriz de Permutación

Una matriz de permutación $\boldsymbol{P}$, es una matriz cuadrada $(n \times n)$ que tiene un 1 en cada fila y columna, el resto de entradas están compuestas por ceros. Por ejemplo:

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} b_3 \\ b_1 \\ b_2 \end{bmatrix}$$

Siempre se cumple que una matriz de permutación es no singular. Este tipo de matrices son de gran ayuda cuando en la diagonal principal se presentan ceros, por lo que se necesita permutar la matriz original pre-multiplicándola por una matriz de permutación. Cuando se pre-multiplica un sistema lineal $\boldsymbol{A}$ por una matriz de permutación $\boldsymbol{P}$, la solución al sistema modificado está dada por la siguiente ecuación:

$$x = (AP)^{-1}b = P^{-1}A^{-1}b = P^{\top}(A^{-1}b)$$

3.1.3. Matriz de Eliminación

Dado un vector **a**, se pueden transformar en cero todas las entradas debajo de una posición k mediante la siguiente transformación:

$$M_k a = \begin{bmatrix} 1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 & \cdots & 0 \\ 0 & \cdots & -m_{k+1} & 1 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & -m_n & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_k \\ a_{k+1} \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} a_1 \\ \vdots \\ a_k \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

donde $m_i = \frac{a_i}{a_k}$, $i = k+1, \ldots, n$.

3.1.4. Matrices Triangulares

El escalado diagonal también es conocido como otro tipo de transformación para matrices. En principio el escalado no cambia la solución del sistema, pero puede afectar el procedimiento numérico para encontrar la solución. Cuando se pre-multiplica, se escalan las filas. Cuando se pos-multiplica, se escalan las columnas. La solución al sistema escalado $\boldsymbol{ADx = b}$ está dado por:

$$x = (AD)^{-1}b = D^{-1}A^{-1}b$$

3.1.5. Método de Sustitución Sucesiva hacia atrás

Para un sistema triangular superior $\boldsymbol{Ax = b}$, es decir, con

$$A = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} \\ 0 & \alpha_{22} & \alpha_{23} \\ 0 & 0 & \alpha_{33} \end{bmatrix}$$

se tiene que la solución es iterativa de la siguiente forma:

$$x_n = \frac{b_n}{\alpha_{nn}}$$

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} \alpha_{ij} x_j}{\alpha_{ii}}, i = n-1, \ldots, 1$$

3.1.6. Método de Sustitución Sucesiva hacia adelante

Para un sistema triangular inferior $\boldsymbol{Ax = b}$, es decir, con

$$A = \begin{bmatrix} \alpha_{11} & 0 & 0 \\ \alpha_{21} & \alpha_{22} & 0 \\ \alpha_{31} & \alpha_{32} & \alpha_{33} \end{bmatrix}$$

se tiene que la solución es iterativa de la siguiente forma:

$$x_1 = \frac{b_1}{\alpha_{11}}$$

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} \alpha_{ij} x_j}{\alpha_{ii}}, i = 2, \ldots, n$$

3.1.7. Método de Eliminación Gaussiana

Para la solución de un sistema de ecuaciones lineales $Ax = b$, el método de eliminación de Gauss propone convertir la matriz (singular) **A** en una matriz triangular superior, esto con el fin de que se pueda resolver mediante el método de sustitución sucesiva hacia atrás.

Para esto, se construye una matriz de eliminación (Ver sección 3.1.3) que permita eliminar las entradas bajo las celdas de la diagonal principal. Este proceso se realiza iterativamente con todas las columnas de la matriz A (con una matriz de eliminación M_i para cada una) de la siguiente manera:

$$MAx = M_{n-1} \ldots M_1 Ax = M_{n-1} \ldots M_1 b = Mb$$

Cuando **A** sea una matriz triangular superior, entonces el problema se puede resolver mediante sustitución sucesiva hacia atrás.

3.1.8. Método de Eliminación Gauss-Jordan

La eliminación Gauss-Jordan es una variación del método de eliminación Gaussiana, solo que aquí en vez de reducirse a un sistema triangular, se hace a un sistema diagonal, es decir, se hacen transformaciones (con el mismo tipo de aniquilación de columnas) hasta conseguir que todas las entradas sean ceros excepto la diagonal. La matriz de eliminación queda de la siguiente forma:

$$\begin{bmatrix} 1 & \cdots & 0 & -m_1 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & -m_{k-1} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & -m_{k+1} & 1 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & -m_n & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_{k-1} \\ a_k \\ a_{k+1} \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ a_k \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

donde $m_i = \frac{a_i}{a_k}$, $i = 1, \cdots, n$.

3.2. Resultados de las simulaciones

3.2.1. Método de Sustitución Sucesiva hacia atrás

- **Ejemplo 1:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 4×4 triangular superior:

$$\begin{bmatrix} 6 & -2 & 2 & 4 \\ 0 & -4 & 2 & 2 \\ 0 & 0 & 2 & -5 \\ 0 & 0 & 0 & -3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 12 \\ 10 \\ -9 \\ 3 \end{bmatrix}$$

  Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} 1 & -3 & -2 & 1 \end{bmatrix}^\top$$

- **Ejemplo 2:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 5×5 triangular superior:

$$\begin{bmatrix} 4 & 8 & 3 & 2 & 9 \\ 0 & 1 & 6 & 3 & 4 \\ 0 & 0 & 9 & 2 & 5 \\ 0 & 0 & 0 & 7 & 2 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 4 \\ 7 \\ 3 \\ 8 \\ 5 \end{bmatrix}$$

  Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} -12{,}6 & 2{,}14 & -2{,}38 & -0{,}28 & 5{,}0 \end{bmatrix}^\top$$

- **Ejemplo 3:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 6×6 triangular superior:

$$\begin{bmatrix} 7 & 20 & 13 & 25 & 89 & 16 \\ 0 & 5 & 56 & 14 & 77 & 6 \\ 0 & 0 & 17 & 15 & 10 & 5 \\ 0 & 0 & 0 & 32 & 8 & 4 \\ 0 & 0 & 0 & 0 & 9 & 2 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 8 \\ 19 \\ 50 \\ 3 \\ 24 \\ 10 \end{bmatrix}$$

  Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} 97{,}13 & -44{,}82 & 1{,}53 & -0{,}80 & 1{,}92 & 3{,}33 \end{bmatrix}^\top$$

3.2.2. Método de Sustitución Sucesiva hacia adelante

- **Ejemplo 1:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 4×4 triangular inferior:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 12 & 4 & 0 & 0 \\ 8 & 5 & 6 & 0 \\ 6 & 15 & 32 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \\ 7 \\ 4 \end{bmatrix}$$

Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} 3{,}0 & -8{,}5 & 4{,}25 & -22{,}5 \end{bmatrix}^\top$$

- **Ejemplo 2:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 5×5 triangular inferior:

$$\begin{bmatrix} -0{,}1 & 0 & 0 & 0 & 0 \\ 15 & 4{,}4 & 0 & 0 & 0 \\ 0{,}8 & 25{,}2 & 26 & 0 & 0 \\ -6 & 1{,}5 & 3{,}2 & 1 & 0 \\ -2 & 0{,}5 & -3{,}5 & 7{,}4 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 12 \\ 0{,}2 \\ 7 \\ 0{,}14 \\ 1 \end{bmatrix}$$

Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} -120{,}0 & 409{,}14 & -392{,}58 & -77{,}28 & -207{,}62 \end{bmatrix}^\top$$

- **Ejemplo 3:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 6×6 triangular inferior:

$$\begin{bmatrix} 15 & 0 & 0 & 0 & 0 & 0 \\ -3 & 19{,}7 & 0 & 0 & 0 & 0 \\ -1{,}5 & 6 & 4 & 0 & 0 & 0 \\ 6 & -1{,}5 & 6 & 7{,}4 & 0 & 0 \\ -1{,}5 & 6 & -1{,}5 & -3 & 10 & 0 \\ -3 & -1{,}5 & 6 & -19{,}7 & -8 & 1{,}5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 15 \\ 19{,}7 \\ 4 \\ 7{,}4 \\ 10 \\ 1{,}5 \end{bmatrix}$$

Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} 1{,}0 & 1{,}15 & -0{,}35 & 0{,}71 & 0{,}62 & 18{,}18 \end{bmatrix}^\top$$

3.2.3. Método de Gauss

- **Ejemplo 1:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz 4×4:

$$\begin{bmatrix} 5 & 2 & 7 & 4 \\ 2 & 5 & 1 & 2 \\ 8 & 4 & 6 & -1 \\ -1 & 2 & -2 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 4 \\ -2 \\ 30 \\ 21 \end{bmatrix}$$

  Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} 16{,}28 & -6{,}32 & -11{,}76 & 4{,}39 \end{bmatrix}^\top$$

- **Ejemplo 2:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz 5×5:

$$\begin{bmatrix} 2 & -1 & 4 & 1 & -1 \\ -1 & 3 & -2 & -1 & 2 \\ 5 & 1 & 3 & -4 & 1 \\ 3 & -2 & -2 & -3 & 1 \\ -4 & -1 & -5 & 3 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 7 \\ 1 \\ 33 \\ 24 \\ 49 \end{bmatrix}$$

  Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} 23{,}13 & 12{,}60 & -12{,}91 & 10{,}5 & -14{,}51 \end{bmatrix}^\top$$

- **Ejemplo 3:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 6×6:

$$\begin{bmatrix} 3 & 10 & 15 & 45 & 22 & 97 \\ -13 & 56 & 70 & -23 & 60 & 5 \\ 8 & 14 & 0 & -45 & 41 & 21 \\ 73 & -12 & 0 & 0 & 61 & 96 \\ 12 & -23 & 0 & 67 & 19 & 10 \\ -64 & 32 & 51 & 73 & -24 & 34 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 37 \\ 71 \\ 11 \\ 42 \\ 94 \\ 96 \end{bmatrix}$$

  Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} -3{,}90 & -9{,}81 & 5{,}95 & -1{,}91 & 1{,}72 & 1{,}09 \end{bmatrix}^\top$$

3.2.4. Método de Gauss-Jordan

- **Ejemplo 1:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 4×4:

$$\begin{bmatrix} 3 & -2 & -1 & -1 \\ -2 & -1 & 2 & 1 \\ 5 & -2 & -2 & -1 \\ 1 & 1 & 2 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -6 \\ 17 \\ -14 \\ 12 \end{bmatrix}$$

  Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} -3{,}0 & -4{,}0 & 2{,}0 & 3{,}0 \end{bmatrix}^\top$$

- **Ejemplo 2:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 5×5:

$$\begin{bmatrix} 2 & -1 & 4 & 1 & -1 \\ -1 & 3 & -2 & -1 & 2 \\ 5 & 1 & 3 & -4 & 1 \\ 3 & -2 & -2 & -2 & 3 \\ -4 & -1 & -5 & 3 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 7 \\ 1 \\ 33 \\ 24 \\ -49 \end{bmatrix}$$

  Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} 1{,}0 & -2{,}0 & 3{,}0 & -4{,}0 & 5{,}0 \end{bmatrix}^\top$$

- **Ejemplo 3:** Dado el siguiente sistema de ecuaciones lineales $Ax = b$, donde A es una matriz (de coeficientes) 6×6:

$$\begin{bmatrix} 13 & 20 & 25 & 55 & 32 & 107 \\ -3 & 66 & 80 & -13 & 70 & 15 \\ 18 & 4 & 5 & -5 & 1 & 2 \\ 3 & -2 & 0 & 0 & 65 & 55 \\ 87 & 9 & 0 & 0 & 9 & 1 \\ -4 & 3 & 5 & 7 & 4 & 25 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 8 \\ 40 \\ 22 \\ 53 \\ 60 \\ 79 \end{bmatrix}$$

  Se tiene como resultado de x el siguiente vector (aproximadamente):

$$\begin{bmatrix} -3{,}21 & 42{,}05 & -33{,}54 & -13{,}38 & -5{,}27 & 8{,}90 \end{bmatrix}^\top$$

3.3. Discusión y análisis

Complejidad del método de Sustitución Sucesiva hacia atrás

Al analizar la función `sucesiva_hacia_atras`, se puede apreciar que su complejidad temporal está en el orden de $O(n^2)$, esto debido a los dos ciclos anidados. Gracias a los límites de los ciclos, se puede notar que se genera un total de $\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = \frac{n^2+n}{2}$ operaciones dentro del cuerpo de la función.

Complejidad del método de Sustitución Sucesiva hacia adelante

Al analizar la función `sucesiva_hacia_adelante`, se puede apreciar que su complejidad temporal está en el orden de $O(n^2)$, esto debido a los dos ciclos anidados. Gracias a los límites de los ciclos, se puede notar que se generan un total de $\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = \frac{n^2+n}{2}$ operaciones dentro del cuerpo de la función.

Complejidad del método de Eliminación de Gauss

Para el método de eliminación Gaussiana, se debe tener cuidado al momento de calcular su complejidad algorítmica, ya que puede que el elemento *k-ésimo* de la diagonal sea cero, por lo cual se debe permutar. Para este análisis se considera siempre el peor caso.

Si se analiza cada una de las funciones y pasos usados para el método de Gauss, se puede apreciar de que se hacen, aproximadamente, un total de $\frac{n(n+1)}{2}$ divisiones, $\frac{2n^3+3n^2-5n}{6}$ multiplicaciones y $\frac{2n^3+3n^2-5n}{6}$ restas. Para un total, aproximado, de $\frac{2n^3}{3}$ operaciones. Según la teoría, se obtiene una complejidad temporal del orden de $O(n^3)$.

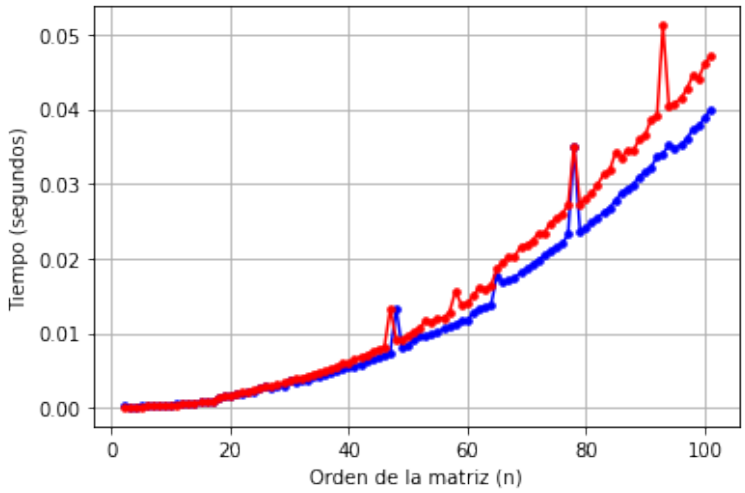
Complejidad del método de Eliminación de Gauss-Jordan

Al analizar la función `gauss_jordan`, que resuelve un sistema de ecuaciones lineales dado por una matriz de coeficientes de tamaño $n \times n$, se puede notar

de manera rápida, que su complejidad está dada por $O(n^3)$, algo parecido al método de eliminación de Gauss.

Complejidades de Gauss vs Gauss-Jordan

En la gráfica que se presenta a continuación se puede observar cómo para una matriz $n \times n$ el algoritmo de `gauss_jordan` (rojo) toma más tiempo en ejecutar la solución que el algoritmo de `gauss` (azul), para $n = 2 \dots 101$:



Bajo la teoría son prácticamente iguales, es decir, $O(\frac{2n^3}{3}) = O(n^3)$. Mientras que, experimentalmente, se puede notar que la gráfica de tiempo de Gauss-Jordan se encuentra por encima de la de Gauss. Esto se debe a que el coeficiente que acompaña al número de operaciones que realiza Gauss es $\frac{2}{3}$, lo cual hace que el n^3 del número total de operaciones se disminuya en $\frac{1}{3}$.

4. Conclusiones

El uso de herramientas computacionales es indispensable en el cálculo de fenómenos científicos como alternativa a los que se realizan de manera clásica, dando la oportunidad de resolver problemas que en la práctica llegarían a ser demasiado tediosos y complejos para que un humano los realice sin margen de error.

Se logró comprender las distintas formas de resolver sistemas de ecuaciones lineales que son bastantes cercanos a la realidad, como también el análisis de la complejidad computacional de cada una de ellas. Por otro lado, en el caso de los métodos de eliminación Gauss y Gauss-Jordan se pudo apreciar la diferencia en cuanto a complejidad computacional se refiere.

5. Referencias

- Material del curso, disponible en BlackBoard
- Bornemann, F., 2016. *Numerical linear algebra*. 1st ed. *Simson, W.*
- Mathews, J., Fink, K., Fernández Carrión, A. & Contreras Márquez, M., 2011. *Métodos Numéricos con MATLAB*. 3rd ed. Madrid: *Pearson Prentice Hall*.
- Librería Numpy
- Librería Pyplot (Matplotlib)
- Librería Time