UNIVERSIDAD TECNOLOGICA NACIONAL

Tecnicatura Universitaria en Programación - Universidad Tecnológica Nacional

TRABAJO PRÁCTICO INTEGRADOR

Programación I

Circuitos RLC y respuesta en Frecuencia e investigación aplicada en Python Análisis de Algoritmos y Estructuras de Datos en circuitos electrónicos

Alumnos:

Gonzalez Javier - Javiergonzalez@fatimarem.edu.ar

Gomez Saucedo Augusto Nahuel - gomeznahuel@hotmail.com

Docente Titular:

Ariel Enferrel

Docente Tutor:

Ramiro Hualpa

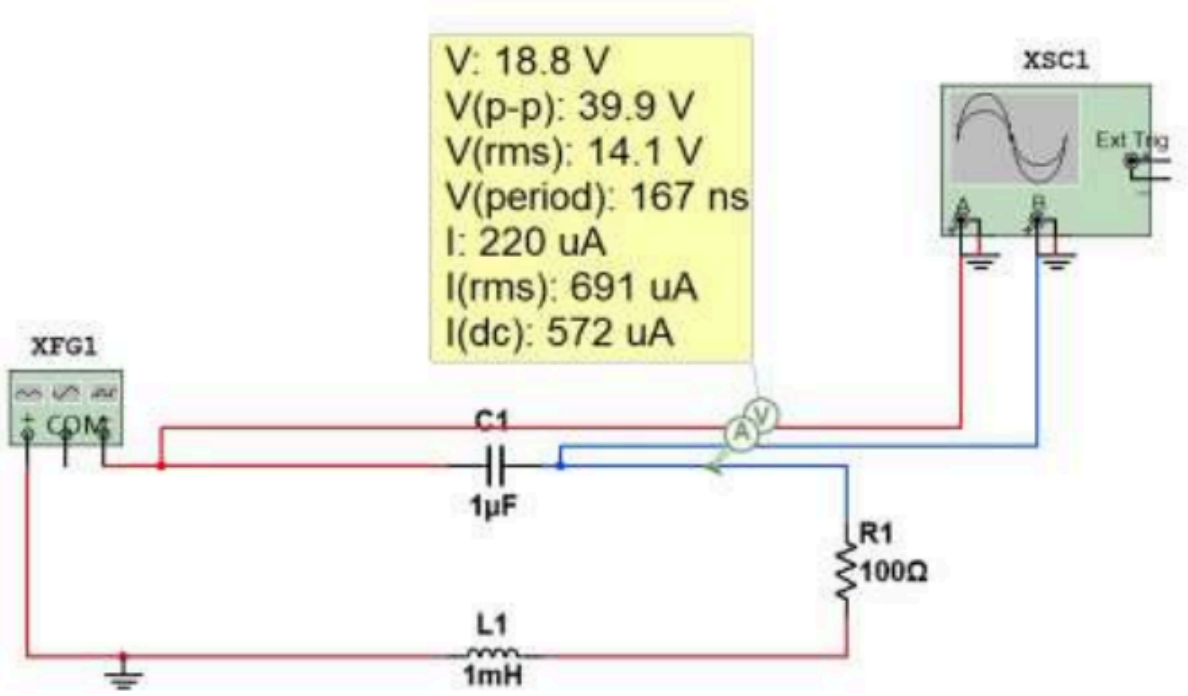
Fecha de Entrega 09 de Junio 2025

Índice

1. Introducción

Los circuitos RLC, compuestos por resistencias (R), inductancias (L) y capacitancias (C), son fundamentales en el estudio de sistemas eléctricos y electrónicos. Su comportamiento frente a señales alternas (AC) varía en función de la frecuencia, lo que los hace esenciales en aplicaciones como filtros, osciladores y sistemas de transmisión.

El análisis en frecuencia permite comprender cómo la impedancia del circuito cambia con la frecuencia de la señal aplicada, lo cual es clave para diseñar sistemas que operen eficientemente en un rango específico del espectro.



2. Marco Teórico Un circuito RLC en serie está compuesto por una resistencia, una bobina y un condensador conectados en línea.

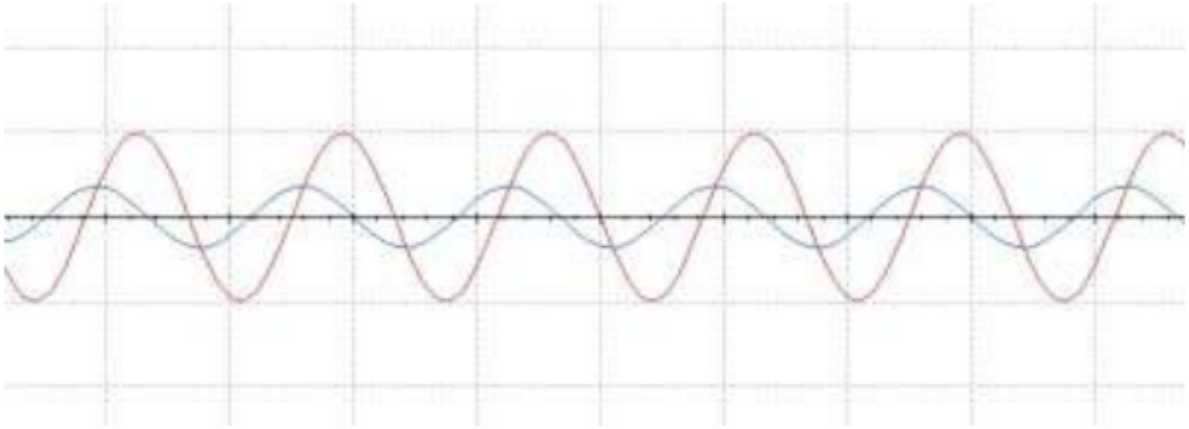


Ilustración 1 señal entrada y señal de salida

La impedancia total del circuito se expresa como:

$Z(\omega)=R+j\omega L+1/j\omega CZ(\omega)$

donde:

- R es la resistencia en ohmios (Ω)
- L es la inductancia en henrios (H)
- C es la capacitancia en faradios (F)
- $\omega=2\pi f\omega=2\pi f$ es la frecuencia angular en radianes por segundo.

La señal de color roja, representa la entrada y la azul la señal de salida. La magnitud de la impedancia determina la oposición total al paso de corriente, mientras que la fase indica el desfase entre la tensión y la corriente.

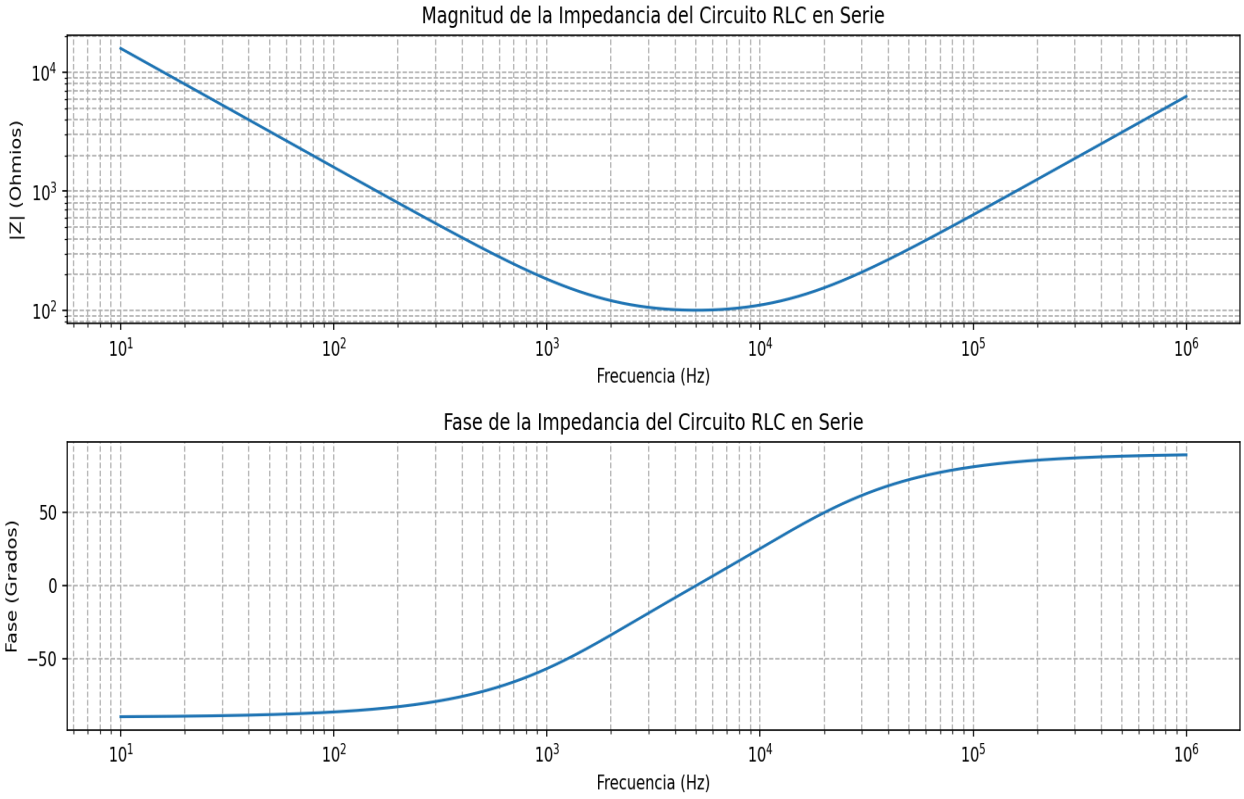
3. Caso práctico

Análisis Computacional con Python

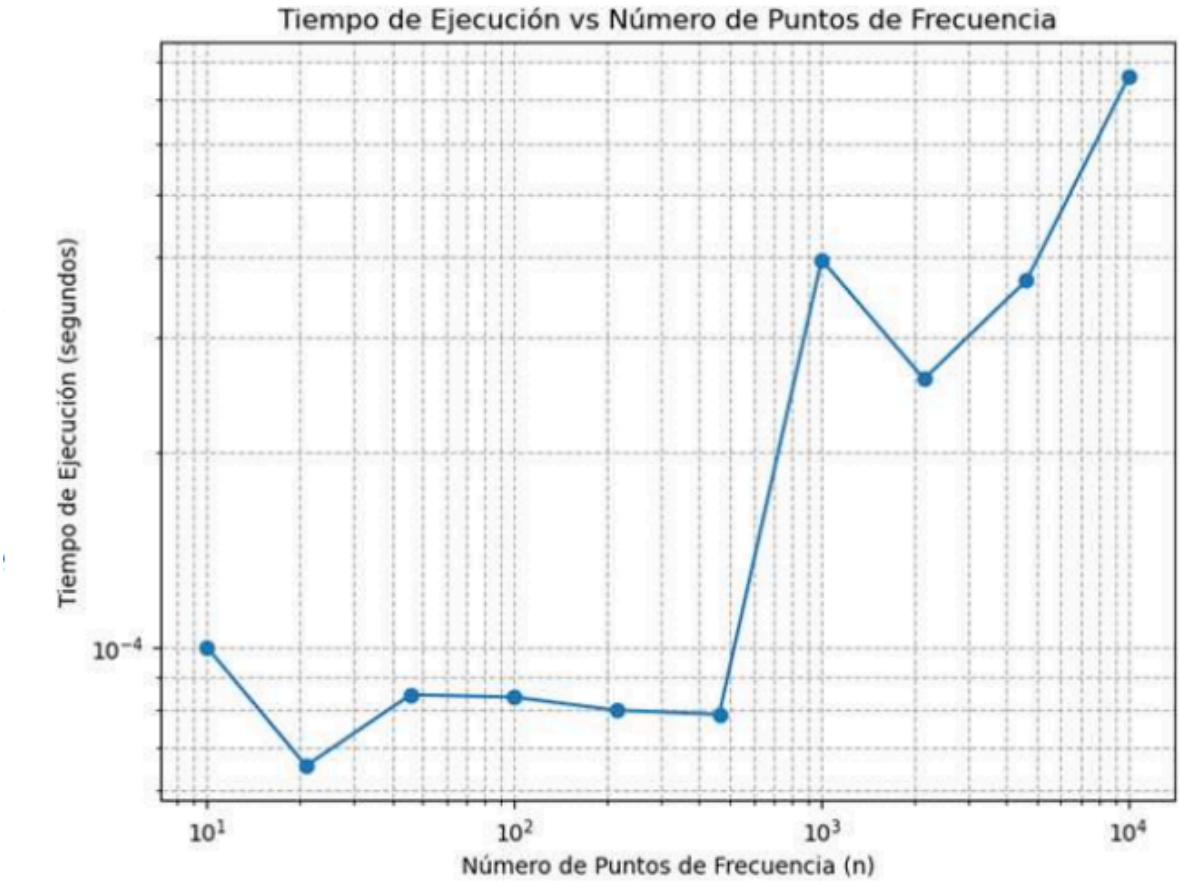
Se construye un programa Python para el análisis numérico y visualización de datos. Utilizando bibliotecas como NumPy y Matplotlib, es posible simular el comportamiento de un circuito RLC en un amplio rango de frecuencias.

El programa desarrollado permite:

- Calcular la impedancia total del circuito para un conjunto de frecuencias.
- Graficar la impedancia y la fase de la impedancia.
- Medir el tiempo de ejecución del análisis para evaluar su eficiencia computacional (complejidad temporal).



El análisis se realiza sobre un conjunto de n frecuencias. Las operaciones vectorizadas de NumPy permiten que el tiempo de ejecución crezca linealmente con n, es decir, la complejidad temporal es: $O(n)$



Esto significa que el programa es eficiente y escalable para simulaciones de alta resolución.

- El gráfico está en escala log-log, lo que permite visualizar mejor el crecimiento.
- La relación es aproximadamente lineal, lo que confirma que el programa tiene una complejidad de tiempo lineal: O(n)

Esto significa que, si se duplica la cantidad de puntos de frecuencia, el tiempo de ejecución también se duplica aproximadamente.

4. Metodología Utilizada

Como primera medida antes de iniciar el código se instalará desde la terminal de Windows la librería que permite utilizar el ploteo de las señales. Se importa la biblioteca NumPy y le damos el alias np . NumPy es una biblioteca muy utilizada en Python para trabajar con matemáticas, álgebra lineal y arreglos para vectores, matrices y sistemas de ecuaciones, esto nos permite aplicar la ecuación algoritmo del circuito RLC.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\Javier> import numpy as np
```

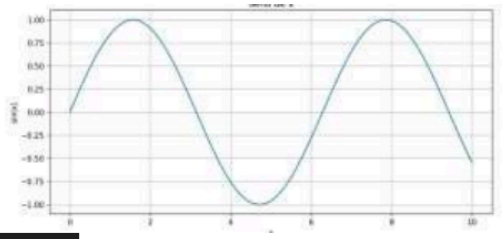
```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\Javier> import matplotlib.pyplot as plt
```

El módulo pyplot de la biblioteca Matplotlib y le damos el alias plt. pyplot se usa para crear gráficos y visualizaciones. A continuación, se realiza un código ejemplo para graficar una función senoidal.

Si la función a graficar es $f(x) = sen\ (x)$

Untitled-1.py > ...

```
1
2   import numpy as np
3   import matplotlib.pyplot as plt
4
5    # Crea 100 puntos entre 0 y 10
6   x = np.linspace(0, 10, 100)
7   y = np.sin(x) # Calcula el seno de cada punto
8   # Dibuja la gráfica
9   plt.title("Seno de x")
10  plt.plot(x, y)
11  plt.xlabel("x")
12  plt.ylabel("sin(x)")
13  plt.grid(True)
14  plt.show()
15
```

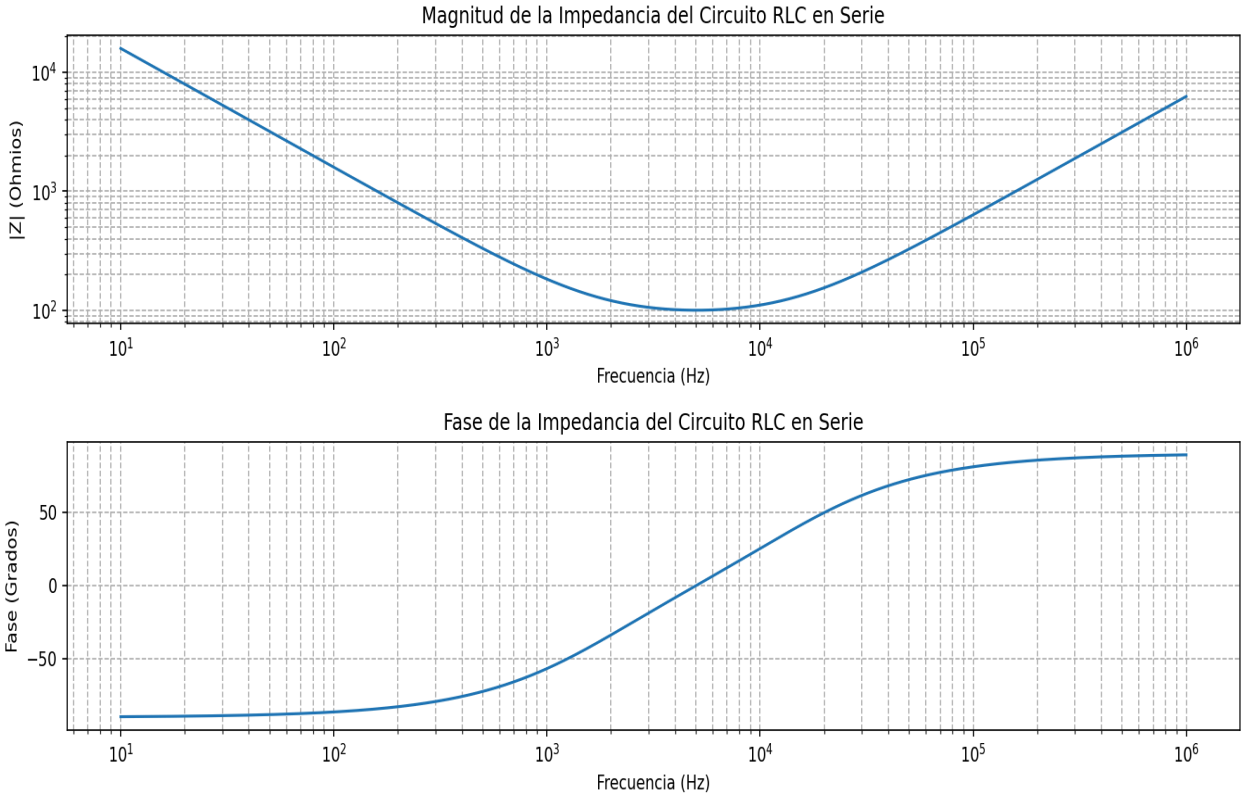


5. Resultados obtenidos

Magnitud y Fase de la Impedancia

La primera parte del análisis muestra cómo varía la impedancia total del circuito en función de la frecuencia:

- La magnitud de la impedancia presenta un mínimo en la frecuencia de resonancia.
- La fase cambia de positiva (inductiva) a negativa (capacitiva), cruzando por 0° en la resonancia.



Se utilizó la fórmula de la impedancia total como el algoritmo principal de un circuito RLC en serie:

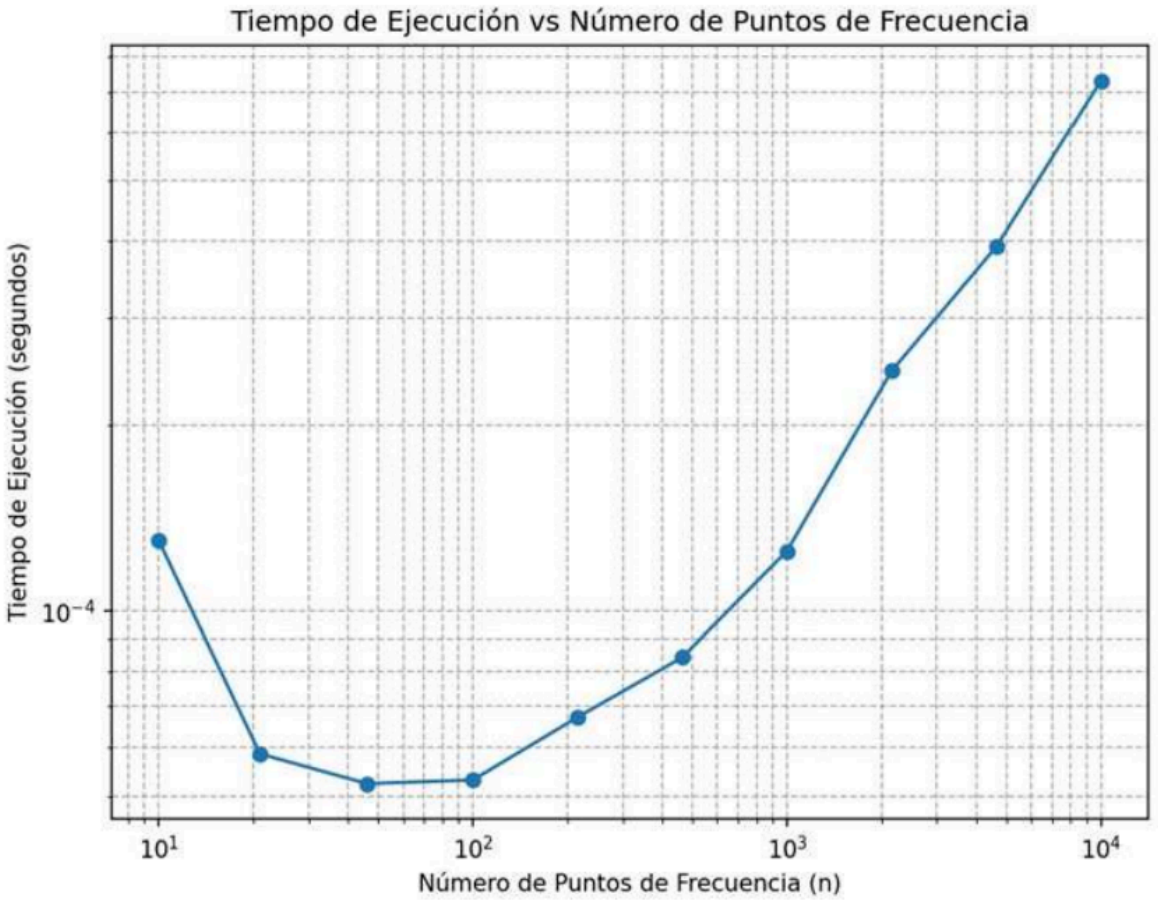
$$Z(w) = R + jwL + (1 jwC)$$

Esta expresión fue implementada en Python usando operaciones vectorizadas con NumPy, lo que permite calcular la impedancia para miles de frecuencias de forma eficiente.

Se generó un rango de frecuencias (de 10 Hz a 1 MHz) y se calculó:

- La magnitud de la impedancia: $|Z|$
- La fase de la impedancia: arc tag (Z)

Este gráfico muestra cómo crece el tiempo de ejecución del análisis a medida que se incrementa el número de puntos de frecuencia (n). La relación es aproximadamente lineal en escala log-log, lo que confirma una complejidad temporal de orden O(n).



Esto permitió observar cómo el circuito responde a diferentes frecuencias, incluyendo el punto de resonancia.

Se usó Matplotlib para graficar:

- La magnitud de la impedancia vs frecuencia.
- La fase de la impedancia vs frecuencia.
- El tiempo de ejecución del análisis para distintos tamaños de entrada (n puntos de frecuencia)

```
# Medición del tiempo de ejecución para distintos valores de n
tiempos_ejecucion = []
valores_n = np.logspace(1, 4, 10, dtype=int)

for n in valores_n:
  inicio = time.time()
  frecuencia = np.logspace(1, 6, n)
  omega = 2 * np.pi * frecuencia
  Z_L = 1j * omega * L
  Z_C = 1 / (1j * omega * C)
  Z_total = Z_R + Z_L + Z_C
  magnitud_Z = np.abs(Z_total)
  fase_Z = np.angle(Z_total, deg=True)
  fin = time.time()
  tiempos_ejecucion.append(fin - inicio)
```

Con el código que se muestra en la ilustración 2, Se mide el tiempo que tarda el programa en ejecutarse para diferentes cantidades de datos (n). Esto permitió verificar que el tiempo de ejecución crece de forma lineal con el tamaño de entrada, lo que indica una complejidad temporal O(n).

6. Conclusiones El análisis computacional permitió observar cómo varía la impedancia de un circuito RLC en serie en función de la frecuencia. Se identificó claramente el fenómeno de resonancia, donde la impedancia alcanza un mínimo y la fase se anula, lo cual es fundamental en el diseño de filtros y sistemas de sintonización. Las gráficas generadas permitieron interpretar de manera intuitiva el comportamiento del circuito, lo cual es especialmente útil en contextos educativos o de diseño. La representación visual de la magnitud y fase de la impedancia ayuda a entender conceptos como la reactancia inductiva y capacitiva.

El análisis del tiempo de ejecución mostró que el programa tiene una complejidad temporal lineal O(n). Esto significa que el tiempo de cálculo crece proporcionalmente al número de puntos de frecuencia analizados, lo cual es eficiente y escalable para simulaciones de alta resolución.

7. Bibliografía

- Repositorio GitHub: Análisis de Circuitos con Python. Ejemplos para el análisis de circuitos eléctricos (incluidos RLC) usando Python.
- Análisis y Simulación de la Transformada de Laplace en Circuito RLC" – Universidad Politécnica Salesiana. Tesis de grado que aborda el análisis de circuitos RLC desde un enfoque teórico y computacional, incluyendo simulaciones con software especializado.
- Análisis introductorio de circuitos, Libro de Robert L. Boylestad. Teoría de circuitos – Boylestad Editorial PEARSON.