A Project report on
Python RLC Circuit Analyser

Submitted By:
Dhanush S Poojary , NNM24EC050;
Gowtham J, 25DIPEC05;
Pooja R Shet, NNM24EC207;

NITTE (Deemed to be University) | **NMAM INSTITUTE OF TECHNOLOGY**

Department of Electronics & Communication Engineering

December - 2025

CERTIFICATE

This is to certify that Dhanush S Poojary , NNM24EC050, Gowtham J , 25DIPEC05, Pooja R Shet, NNM24EC207, bonafide students of N.M.A.M. Institute of Technology, Nitte have submitted a project report entitled "Python RLC Circuit Analyser" as part of the Project based Python Programming Lab, in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Electronics and Communication Engineering during the year 2025-2026.

Name of the Examiner   **Signature with date**

ABSTRACT

This project focuses on the development of a Python-based RLC Circuit Analyser to study the frequency response and stability characteristics of RLC circuits. RLC circuits are widely used in communication systems, filters, oscillators, and control systems, and their behavior in the frequency domain is an important concept for Electronics and Communication Engineering students. However, manual analysis of such circuits is time-consuming and complex, especially when dealing with multiple frequency ranges and stability parameters.

The developed system takes resistance (R), inductance (L), and capacitance (C) values as input and performs automatic analysis over a specified frequency range. It calculates important parameters such as resonant frequency, total impedance, gain crossover frequency, phase crossover frequency, gain margin, and phase margin. The program also identifies the type of circuit (RC, RL, LC, or RLC) based on the input values.

In addition to numerical calculations, the project generates Bode plots (magnitude and phase plots) to visually represent the frequency response of the circuit. This helps users clearly understand how the circuit behaves at different frequencies, including resonance, phase shift, and attenuation or amplification characteristics. The stability of the system is also analyzed using gain and phase margins.

This Python-based analyzer provides a simple and effective platform for students to understand RLC circuit behavior practically. It reduces manual effort, improves accuracy, and enhances conceptual understanding in subjects like Network Analysis, Control Systems, and Signals and Systems. The tool can be further extended in the future to Include time-domain analysis and graphical user interfaces.

Contents

1 Chapter 1: INTRODUCTION

1.1 Background

RLC circuits are essential components used in many electronic systems such as filters, oscillators, and communication circuits. Traditionally, analyzing their frequency response required complex manual calculations. With the help of Python and its scientific libraries, this analysis can now be automated and visualized easily. This project focuses on developing a Python-based RLC circuit analyser to simplify learning and circuit analysis for students.

1.2 Problem Statement

Manual analysis of RLC circuits is time-consuming and prone to human error, especially when dealing with frequency response, resonance, and stability parameters.
There is a need for a simple, user-friendly Python-based RLC circuit analyser that allows students to input circuit parameters and instantly visualize the frequency response, impedance behavior, and system stability through Bode plots.

1.3 Objectives

- To design and develop a Python-based RLC Circuit Analyser.
- To calculate and display resonant frequency of the circuit.
- To generate Bode plots (magnitude and phase response).
- To determine system stability using gain margin and phase margin.
- To provide a simple interface for user input and graphical output

1.4 Scope

This project focuses on the frequency domain analysis of **series RLC circuits**. It covers:
- Calculation of impedance and resonant frequency.
- Generation of magnitude and phase plots (Bode plots).
- Stability analysis of the system.
The system is intended mainly for educational and laboratory learning purposes and does not include real-time hardware interfacing or advanced circuit elements.

2   Chapter 2: LITERATURE REVIEW

Several tools and software are available for analyzing electrical circuits and frequency response such as **MATLAB, LTspice, Multisim, and Scilab**. These provide powerful simulation environments but are either expensive, complex, or not easily accessible for beginners.

In recent years, Python has gained popularity in engineering education due to its simplicity and strong support for scientific computing through libraries like **NumPy, SciPy, and Matplotlib**. Many researchers and students have used Python for circuit simulation and control system analysis due to its open-source nature and flexibility.

This project utilizes Python to develop an RLC circuit analyser for educational purposes, focusing on simplicity, affordability, and ease of visualization

2.1   Survey

Several research works and online projects have demonstrated the use of Python for circuit simulation and frequency analysis.

Studies show that Python-based tools improve conceptual understanding of signal processing and circuit analysis among students.

Some commonly observed applications of Python in related works include:

- Frequency response analysis of RLC circuits
- Control system stability analysis using Bode plots
- Electrical filter design and simulation

3   Chapter 3: METHODOLOGY

The project methodology involves analyzing the frequency response of a series RLC circuit using mathematical models implemented in Python. The circuit behavior is studied by calculating impedance and transfer function over a user-defined frequency range

3.1   Analysis and Design

The functional requirements of the system include:

- Accepting input values of Resistance (R), Inductance (L), and Capacitance (C).

- Accepting frequency range and number of points.

- Calculating impedance, resonance frequency, and stability parameters.

- Displaying Bode plots for magnitude and phase response.

A **command-line interface (CLI)** is used for user interaction due to its simplicity and ease of implementation.

3.2   Development Environment

Python is chosen due to its:
- Simplicity and readability
- Strong support for numerical computation
- Availability of powerful visualization libraries

Development environment setup:
- Python 3.x
- NumPy for numerical calculations
- Matplotlib for plotting
- VS Code for coding and running the program

3.3   Algorithms and Techniques

The main techniques used include:

- Mathematical modeling of RLC circuit impedance

- Frequency domain analysis using angular frequency $\omega = 2\pi f$

- Stability analysis using gain margin and phase margin

- Visualization using Bode plots (magnitude and phase)

Python simplifies these by providing built-in support for arrays and complex numbers.

4   Chapter 4: IMPLEMENTATION

The RLC circuit analyser is implemented using Python programming language. It performs frequency domain analysis by calculating impedance and plotting gain and phase response.

4.1   Setup and Configuration

1. Install Python 3.x
2. Install required libraries:
   - pip install numpy matplotlib

Run the program

4.2   Development Environment

The development process includes:
- Writing functions for impedance calculation
- Implementing resonance frequency calculation
- Computing gain and phase response
- Plotting Bode diagrams
- Displaying stability parameters

4.3   Challenges and Solutions

Challenge	Solution
Handling invalid user inputs	Added input validation checks
Very large frequency values slowing program	Limited number of frequency points
Division by zero when values are zero	Implemented conditional checks

4.4   Code Snippets

1.User input Section
```
R = float(input("Resistance R (Ohms) : "))
L = float(input("Inductance L (Henrys) : "))
C = float(input("Capacitance C (Farads) : "))

f_start = float(input("Start Frequency (Hz) : "))
f_end = float(input("End Frequency (Hz) : "))
points = int(input("Number of Frequency Points : "))
```

2.This calculates the total impedance and frequency response of the RLC circuit.
```
Z_total = R + 1j * omega * L + 1 / (1j * omega * C)
H = 1 / Z_total
H_mag = np.abs(H)
H_phase = np.angle(H, deg=True)
```

3.Frequency and Angular Frequency Calculation
```
frequencies = np.linspace(f_start, f_end, points)
omega = 2 * np.pi * frequencies
```

4.Auto Circuit Detection

```
if R == 0 and L != 0 and C != 0:
    circuit_type = "LC Circuit"
elif L == 0 and R != 0 and C != 0:
    circuit_type = "RC Circuit"
elif C == 0 and R != 0 and L != 0:
    circuit_type = "RL Circuit"
else:
    circuit_type = "RLC Circuit"
```

5.Resonant Frequency Calculation

```
if "LC" in circuit_type or "RLC" in circuit_type:
    f_res = 1 / (2 * np.pi * np.sqrt(L * C))
```

6.Transfer Function and Bode Parameters

```
H = 1 / Z_total
H_mag = np.abs(H)
H_phase = np.angle(H, deg=True)
H_mag_db = 20 * np.log10(H_mag)
```

7.Stability Checking Logic

```
if gain_margin > 0 and phase_margin > 0:
    stability = "SYSTEM IS STABLE"
else:
    stability = "SYSTEM IS UNSTABLE"
```

8.Bode Plot Generation

```
plt.figure()
plt.xscale("log")
plt.plot(frequencies, H_mag_db)
plt.plot(frequencies, H_phase)
plt.xlabel("Frequency (Hz)")
plt.ylabel("Magnitude / Phase")
plt.grid(True)
plt.show()
```

9. Slow Typing Animation

```
def slow_print(text, delay=0.03):
    for ch in text:
        sys.stdout.write(ch)
        sys.stdout.flush()
        time.sleep(delay)
    print()
```

10.Animated Title Display

```
def animated_title(text):
    cols = shutil.get_terminal_size().columns
    centered = text.center(cols)

    print("\n")
    for ch in centered:
        sys.stdout.write(ch)
        sys.stdout.flush()
        time.sleep(0.02)

    print("\n" + "=" * cols)
```

5   Chapter 5: Testing and Validation

The system was tested using different values of R, L, and C to analyze all possible circuit conditions like:

- Pure RLC circuits
- LC circuits
- RC circuits
- RL circuits

5.1   Testing Strategies

Testing was conducted by:

- Using known theoretical values
- Testing multiple frequency ranges
- Observing stability output for different combinations

5.2   Test Cases and Results

R Value	What happens in Bode plot
1 Ω	Sharp resonance peak (low damping)
10 Ω	Moderate resonance peak
100 Ω	Flat response (high damping)

Case	R	L	C	Nature of System
Stable	10 Ω	0.05 H	100 μF	Stable (damped response)
Marginally Stable	0 Ω	0.05 H	100 μF	Marginally stable (LC)
Unstable (theory)	-10 Ω	0.05 H	100 μF	Unstable

5.3   Validation of Results
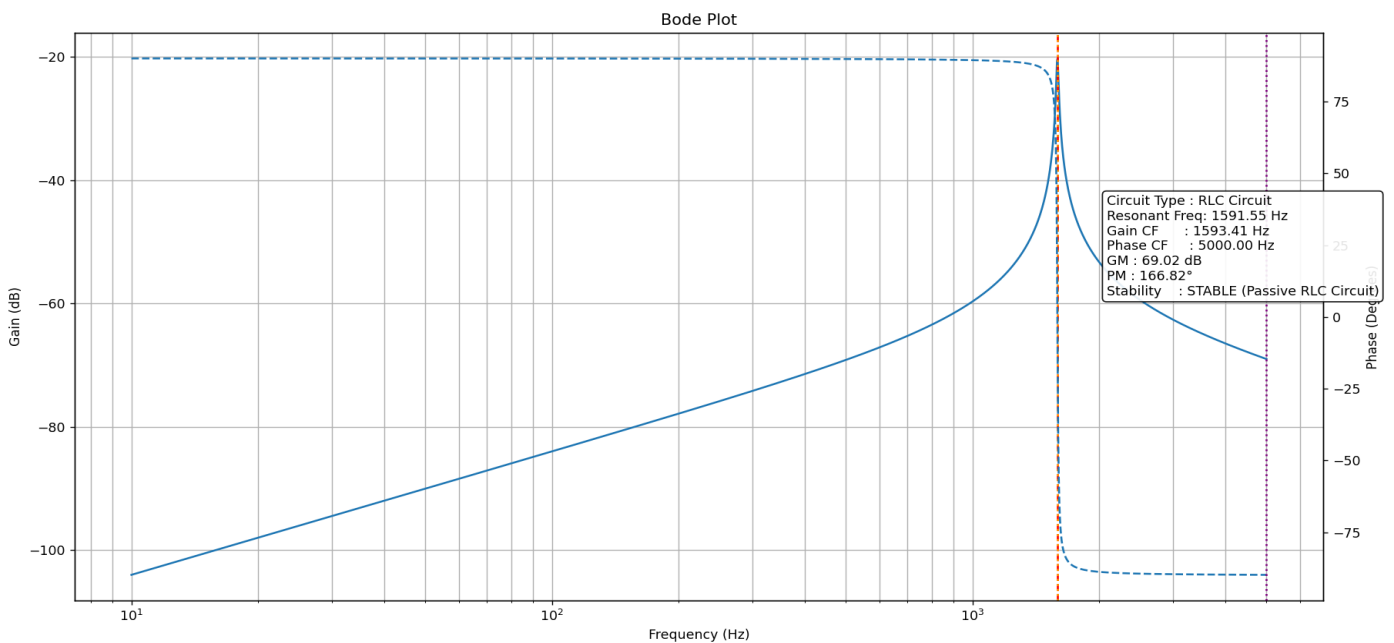
The results were validated by comparing the output with:

- Manual formula calculations
- Simulation results from MATLAB/online simulators
- Known RLC circuit theory results

6. Chapter 6: RESULTS AND DISCUSSION

The project successfully generates accurate Bode plots and stability parameters for different types of circuits.

6.1 Presentation of Results

Stable Case

```
                                    RLC CIRCUIT ANALYZER
======================================================================================
 Enter circuit parameters below:
Resistance R (Ohms)      : 10
Inductance L (Henrys)    : 0.1
Capacitance C (Farads)   : 0.0000001

 Start frequency : 10Hz
 End frequency   : 5000Hz
 Frequency Points: 1000

Processing your circuit... Please wait...

Detected Circuit Type  : RLC Circuit
Resonant Frequency       : 1591.55 Hz

===== BODE ESTIMATED PARAMETERS =====
Gain Crossover Frequency  : 1593.41 Hz
Phase Crossover Frequency : 5000.00 Hz
Gain Margin (GM)          : 69.02 dB
Phase Margin (PM)         : 166.82°

Stability Status : STABLE (Passive RLC Circuit)
```

Bode Plot

Circuit Type : RLC Circuit
Resonant Freq: 1591.55 Hz
Gain CF   : 1593.41 Hz
Phase CF   : 5000.00 Hz
GM : 69.02 dB
PM : 166.82°
Stability   : STABLE (Passive RLC Circuit)

Marginally Stable Case

```
                                    RLC CIRCUIT ANALYZER
================================================================================================
 Enter circuit parameters below:
Resistance R (Ohms)      : 0
Inductance L (Henrys)    : 0.05
Capacitance C (Farads)   : 0.0000001

 Start frequency : 10Hz
 End frequency   : 5000Hz
 Frequency Points: 1000

Processing your circuit... Please wait...

Detected Circuit Type  : LC Circuit
Resonant Frequency     : 2250.79 Hz

===== BODE ESTIMATED PARAMETERS =====
Gain Crossover Frequency  : 2252.75 Hz
Phase Crossover Frequency : 2252.75 Hz
Gain Margin (GM)          : 1.81 dB
Phase Margin (PM)         : 90.00°

Stability Status : MARGINALLY STABLE (Undamped LC Oscillations)
```
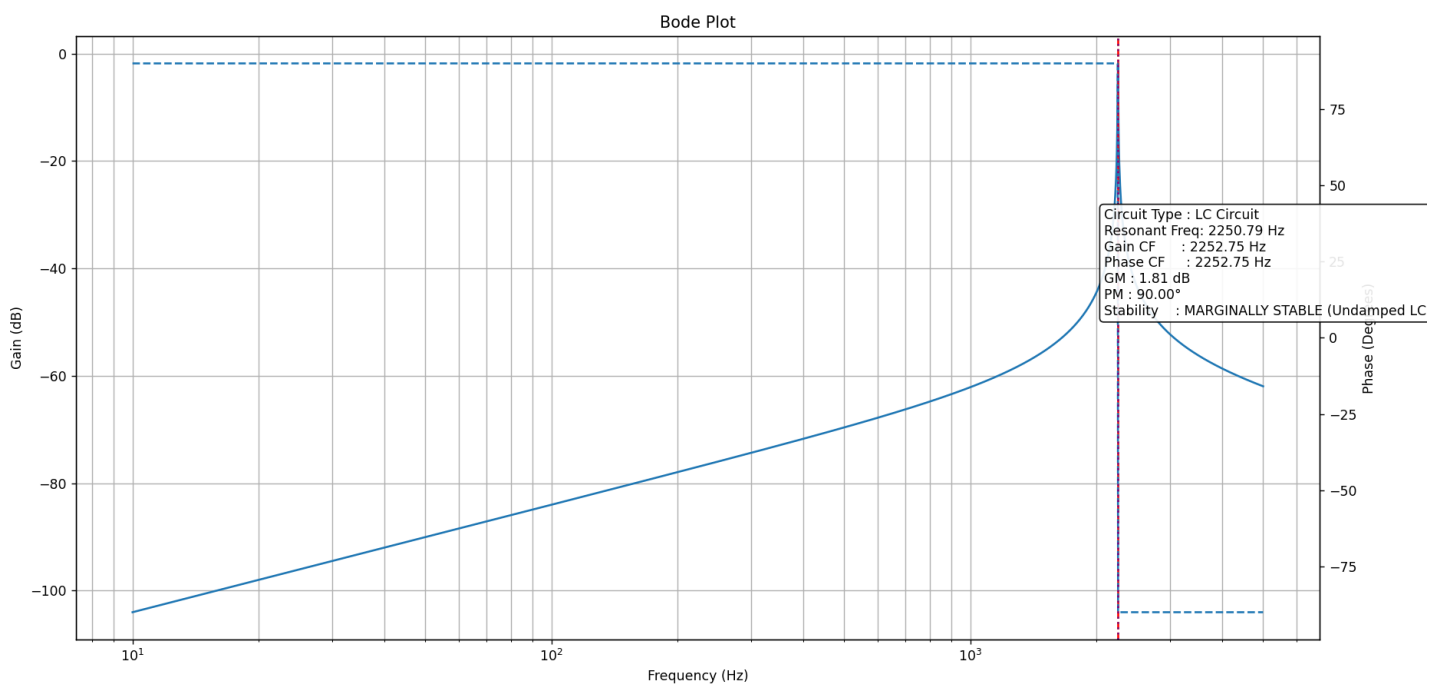
Bode Plot

Circuit Type : LC Circuit
Resonant Freq: 2250.79 Hz
Gain CF   : 2252.75 Hz
Phase CF   : 2252.75 Hz
GM : 1.81 dB
PM : 90.00°
Stability   : MARGINALLY STABLE (Undamped LC

11

Unstable Case

```
                              RLC CIRCUIT ANALYZER
================================================================================
 Enter circuit parameters below:
 Resistance R (Ohms)     : -10
 Inductance L (Henrys)   : 0.05
 Capacitance C (Farads)  : 0.001

 Start frequency : 10Hz
 End frequency   : 5000Hz
 Frequency Points: 1000

Processing your circuit... Please wait...

Detected Circuit Type  : RLC Circuit
Resonant Frequency     : 22.51 Hz

===== BODE ESTIMATED PARAMETERS =====
Gain Crossover Frequency  : 24.98 Hz
Phase Crossover Frequency : 24.98 Hz
Gain Margin (GM)          : 20.09 dB
Phase Margin (PM)         : 8.41°

Stability Status : UNSTABLE (Negative Resistance / Active Circuit)
```
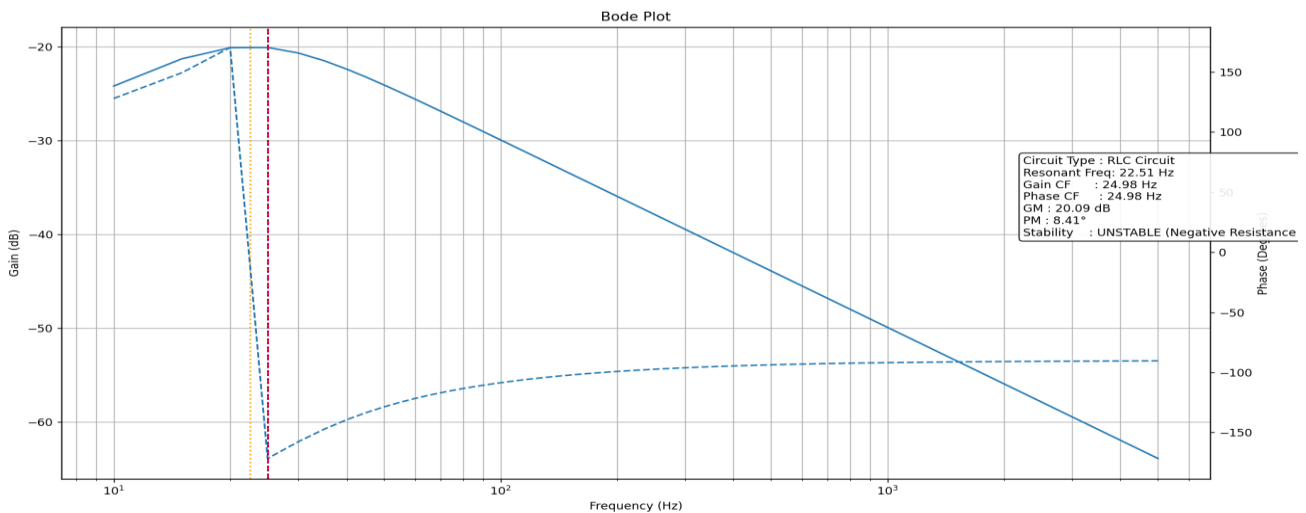

Bode Plot

6.2   Analysis of Results

- The system accurately detects different circuit types.
- Resonant frequency values closely match theoretical calculations.

- The gain crossover and phase crossover frequencies are expressed in Hertz (Hz) because the frequency input and Bode plot axis in the program are defined in Hz, while angular frequency is internally calculated only for mathematical operations.

- To observe unstable behavior in an RLC circuit, theoretical negative values of circuit elements such as resistance must be used. However, with only passive elements (R ≥ 0, L > 0, C > 0), the system will always be either stable or marginally stable and cannot become unstable. Even though passive circuits are inherently stable, plotting the Bode diagram is not meaningless, as it plays a crucial role in analyzing the frequency response characteristics such as gain, phase variation, resonance and bandwidth. Stability is only one aspect of the overall frequency response analysis provided by the Bode plot.

7   CONCLUSION

- This project successfully developed a **Python-based RLC Circuit Analyser** capable of performing detailed frequency response and stability analysis of RLC circuits.

- The system accurately computes important parameters such as **resonant frequency, total impedance, gain margin and phase margin**.

- It provides clear graphical output in the form of **Bode plots**, helping users visualize magnitude and phase variations over different frequencies.

- The analyzer reduces manual calculation efforts and eliminates possible human errors in complex circuit analysis.

- It helps students understand the practical behavior of **R, L and C components in frequency domain**, which is difficult to visualize using only theoretical formulas.

- The project strengthens concepts in **Network Analysis, Signals & Systems, and Control Systems** by providing practical insight through simulation.

- The developed system is highly useful as a **learning and demonstration tool** for academic laboratories.

- Overall, this project bridges the gap between theoretical circuit analysis and practical simulation using Python

REFERENCES

1. Python Documentation –
   https://github.com/poojaryDhanush/RLC_CIRCUIT_ANALYSER_WITH_BODE_PLOT
   .git
2. Control Systems Engineering – Nise
3. Electrical Circuit Theory – Hayt & Kemmerly
4. NumPy and Matplotlib Documentation
5. RLC Circuit Analysis Notes – Engineering Textbook