SOFTWARE IMPLEMENTATION OF PSPISE-BASED MODELS IN THE PYTHON PROGRAMMING LANGUAGE

*Stoyan Popov (1), Nikolay Hinov (2)**

(1) Faculty of Applied Mathematics and Informatics; (2) Faculty of Electronic Engineering and Technologies, Technical University of Sofia, Sofia 1000 Bulgaria

* Corresponding Author, e-mail: hinov@tu-sofia.bg

Abstract: The manuscript presents the software implementation of Pspise-based models in the Python programming language. To demonstrate the model transformation process, examples of electrical circuits that present current and voltage resonance are considered. The implementation experiments were conducted based on models implemented in LTspice. Simulation results using Python are obtained that are similar to those from other software environments specialized for modeling electrical and electronic circuits.

Key words: computer simulations, LTspice, models, Python, software implementation.

1. INTRODUCTION

Advances in computational mathematics, modeling, and automated methods for data extraction and analysis pose new challenges to researchers and designers of electronic devices and systems. In this sense, the integration of modern information and communication technologies in scientific research provides additional opportunities both for designing and prototyping electronic devices and systems with guaranteed indicators, and for their use in training [1,2]. The development of electrical and electronic circuit design is inextricably linked to the application of various modeling and automated design environments. In practice, even with the creation of the first electronic computing machines, attempts to create various programs for simulation research of various schemes began. Due to the topicality and importance of this research issue, both teams of researchers and companies specializing in the field of application software development have worked on it and are working on it. Thus, a variety of programming environments are currently used to model electrical circuits and various electronic devices. [3,4] The most popular circuit modeling software environments that are used by researchers, engineers and

students for simulation, analysis and optimization of various electrical circuits and systems are as follows [5,6]:

- SPICE (Simulation Program with Integrated Circuit Emphasis) is one of the most popular and long-established environments for the simulation of electrical circuits. It is used to analyze various elements of electronic circuits and systems such as transistors, optocouplers, resistors, capacitances and inductances.
- NI Multisim (formerly known as Electronics Workbench) is a commercial electrical and electronic circuit simulation environment that offers a graphical user interface for easy model creation and circuit analysis.
- LTspice is a free circuit simulation environment developed by Linear Technology (after the merger with Analog Devices, now part of Analog Devices). It provides a large set of components and libraries, mostly produced by Analog Devices.
- PSPICE is a commercial circuit simulation environment developed by Cadence Design Systems. It offers powerful functions for simulation and optimization of complex circuit topologies.
- Simulink is a MATLAB package that is used for simulation and modeling of various systems, including electrical circuits.
- QUCS (Quite Universal Circuit Simulator) is a free and open source circuit simulation software. It provides a graphical interface and supports various components and simulation methods.
- Proteus is a software environment that combines circuit simulation with printed circuit board (PCB) design.

Most of these products are SPICE-based environments, which offer powerful capabilities for the analysis and simulation of electrical circuits and can be used for a variety of projects and applications. For this reason, they have become established both in the development of commercial projects and for the training needs of students and specialists.

Python is a high-level programming language that is popular for its readability and easy-to-learn syntax. This language is used for web application development, scientific research, data analysis, artificial intelligence, automation and much more. Python is an extremely popular and versatile programming language with many advantages, but it also has some disadvantages. Below is a summary of some of them [7,8].

Advantages of Python:

- Python has a readable and intuitive syntax that makes code clear and understandable, making it an excellent choice for beginner programmers as it is easy to learn and has a readable syntax.
- Python has a wide range of libraries and modules that cover multiple areas - from web development and scientific computing to artificial intelligence and machine learning. Large community and resources make solving problems easier.

- Python runs on a variety of operating systems, including Windows, macOS, and various Linux distributions, allowing the development of easily portable applications between different operating systems.

- Python can easily be integrated with other programming languages such as C, C++, Java, which allows the use of libraries and functionalities from these languages.

- Python supports both object-oriented and functional programming styles, which gives flexibility in software design.

Unfortunately, Python also has some disadvantages compared to other high-level languages:

- Python's performance can be slower compared to compiled languages like C++ or Java. This problem is more noticeable in applications that require high performance or fast data processing.

- Python can use more memory than some other languages, which can be a problem when working with large data or limited resources.

- In the standard Python interpreter - CPython, the global interpreter lock (GIL) limits the simultaneous execution of threads and can cause difficulties in the optimization of multi-threaded code.

- Despite its flexibility, Python is not the best choice for all types of applications, such as embedded systems with limited resources or tasks that require a low level of close-to-hardware programming.

In this regard, from the point of view of application of computer modeling and simulations, the advantages of Python are very useful, especially in the study of electrical and electronic engineering by software engineering students.

On the other hand, due to the long-term use of Spice-based modeling environments in electrical engineering and electronics, many and the most diverse models of various building blocks, individual circuits and systems have been developed. In practice, many model libraries are available, created by both electronic component manufacturers and researchers. These environments are intuitive and easy to initially learn and work with, as they have a well-developed user interface and also provide a variety of options for visualization of the obtained results. Unfortunately, Spice-based environments are less capable of collecting, analyzing and processing large volumes of data, as well as performing various optimization tasks.

The aim of the present work is to present the implementation of list-based models in Python and thereby provide additional opportunities both for researchers in the field of electronics and electrical engineering, and to support the electrical engineering education of software engineers.

2. CONCEPT OF THE SOFTWARE IMPLEMENTATION OF THE MODELS

In Fig. 1 shows the block diagram that illustrates the individual steps of the proposed algorithm for transforming an LTspice model into a Python executable format.

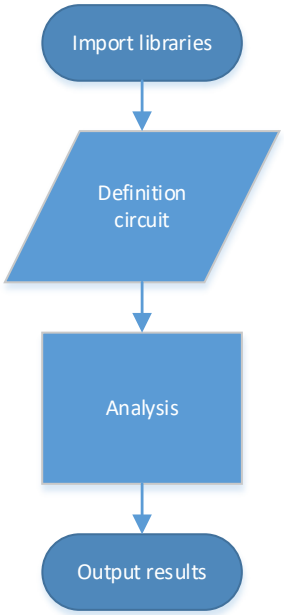


Figure 1. Algorithm flowchart for implementing Spice-based models in Python

The implementation process of Script-based models proceeds in the following sequence:

1. Libraries and modules required for the operation of the program are loaded:

import PySpice.Logging.Logging as Logging – registering the program;

from PySpice.Spice.Library import SpiceLibrary – loading external item libraries;

from PySpice.Spice.Netlist import Circuit – load the class creating the circuit;

from PySpice.Unit import * - loading the library containing the measurement units;

2. Description of the specific electric circuit, object of research:

Specifying the elements involved in a circuit by their names, start and end connection nodes, and their size or type when it comes to a current or voltage source.

3. Perform analysis:

simulator = circuit.simulator() – load the simulator

analysis = simulator.ac – AC Analysis

analysis = simulator.dc – DC analysis

analysis = simulator.transient – Analysis of transient processes

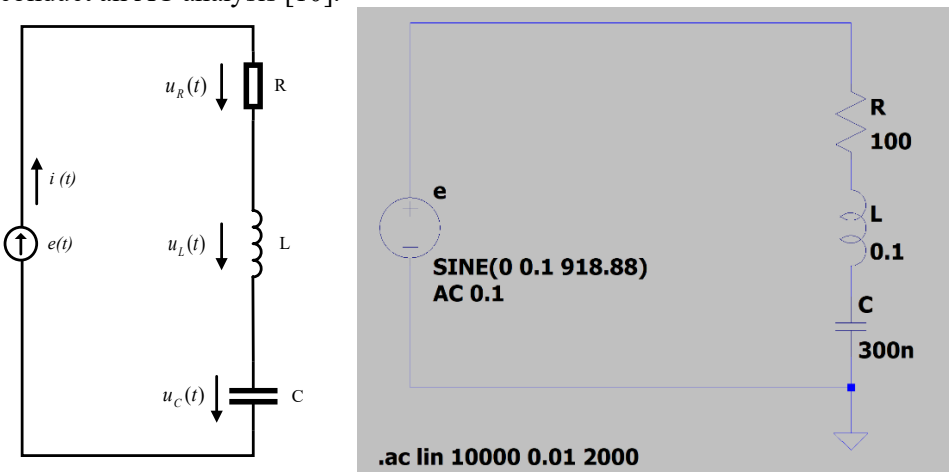
4. Output of the obtained results in text or graphic form.

Of course, the algorithm of the transformation process is given in a general form, and for each specific case, the necessary Python element model library (already existing or added by the user) should be used, as well as the necessary analyzes, according to the goals of the study.

3. EXAMPLES FOR IMPLEMENTATION OF MODELS IN PYTHON

3.1. Scheme of series resonant circuit

The schematic of a series resonant circuit is shown in Fig.2a. It consists of a sinusoidal voltage source *e(t)*, an inductance, a capacitance and a resistor. This computational example is taken from a training manual for theoretical electrical engineering students at TU-Sofia [9]. Fig. 2.b shows the LTspise model of the scheme, with the values of the scheme elements and the parameters of the analysis set. In this case, according to the need to study resonance processes, it was chosen to conduct an AC analysis [10].



a *b*

Figure 2. Series resonant circuit: a) circuit diagram; b) model in LTspice

The scheme and model parameters are:

$e(t) = 0{,}1.\sin(5773{,}5t)$, $R = 100\,\Omega$, $L = 0{,}1H, C = 300nF$.

The implementation of the model in Python was carried out based on the algorithm presented in Fig. 1, and for this purpose an author's program was created. The source code of this program is shown in Fig.3.

```
import PySpice.Logging.Logging as Logging
logger = Logging.setup_logging()

import matplotlib.pyplot as plt
import numpy as np

#import PySpice
from PySpice.Spice.Library import SpiceLibrary
from PySpice.Spice.Netlist import Circuit
from PySpice.Unit import *

circuit = Circuit('Voltage resonance')
```

```
circuit.VoltageSource('e', 1, circuit.gnd, 'AC 0.1 SIN(0 0.1 918.88) external')
circuit.R('R', 1, 2, 100@u_Ohm)
circuit.L('L', 2, 3, 0.1@u_H)
circuit.C('C', 3, circuit.gnd, 300@u_nF)

print(circuit)

simulator = circuit.simulator()
analysis = simulator.ac(variation = 'lin', number_of_points = 10000, start_frequency = 0.01@u_Hz,
stop_frequency = 2000@u_Hz)

figure, ax = plt.subplots(figsize = (12, 6))
ax.plot(analysis.frequency, 20*np.log10(np.absolute(analysis['1'] - analysis['2'])), color = 'red',
linewidth = 2)
ax.plot(analysis.frequency, 20*np.log10(np.absolute(analysis['2'] - analysis['3'])), color = 'green',
linewidth = 2)
ax.plot(analysis.frequency, 20*np.log10(np.absolute(analysis['3'])), color = 'blue', linewidth = 2)
ax.set_xticks(np.arange(0, 2200, 200))
ax.set_yticks(np.arange(-220, 10, 10))
ax.set_title('Frequency response of elements')
ax.set_xlabel('Frequency [Hz]')
ax.set_ylabel('Amplitude [Db]')
ax.legend(('FR(R)', 'FR(L)', 'FR(C)'), loc = (0.8, 0.5))
ax.grid()

figure1, ax1 = plt.subplots(figsize = (12, 6))
ax1.plot(analysis.frequency, np.absolute(analysis['1'] - analysis['2']), color = 'red', linewidth = 2)
ax1.plot(analysis.frequency, np.absolute(analysis['2'] - analysis['3']), color = 'green', linewidth = 2)
ax1.plot(analysis.frequency, np.absolute(analysis['3']), color = 'blue', linewidth = 2)
ax1.set_xticks(np.arange(0, 2200, 200))
ax1.set_yticks(np.arange(0.0, 0.65, 0.05))
ax1.set_title('Voltages across the elements')
ax1.set_xlabel('Frequency [Hz]')
ax1.set_ylabel('Voltage [V]')
ax1.legend(('U(R)', 'U(L)', 'U(C)'), loc = (0.8, 0.8))
ax1.grid()

figure2, ax2 = plt.subplots(figsize = (12, 6))
ax2.plot(analysis.frequency, np.unwrap(np.angle(analysis['1'] - analysis['2'], deg=True)), color = 'red',
linewidth = 2)
ax2.plot(analysis.
```

Figure 3. Source code of the author's program for implementing the LTspice model of a series resonant circuit in Python

In fig. 4 presents the results of the simulation of a series resonant circuit in Python. Specifically, the dependences of the voltages on the three elements of the series RLC circuit as a function of frequency are given.

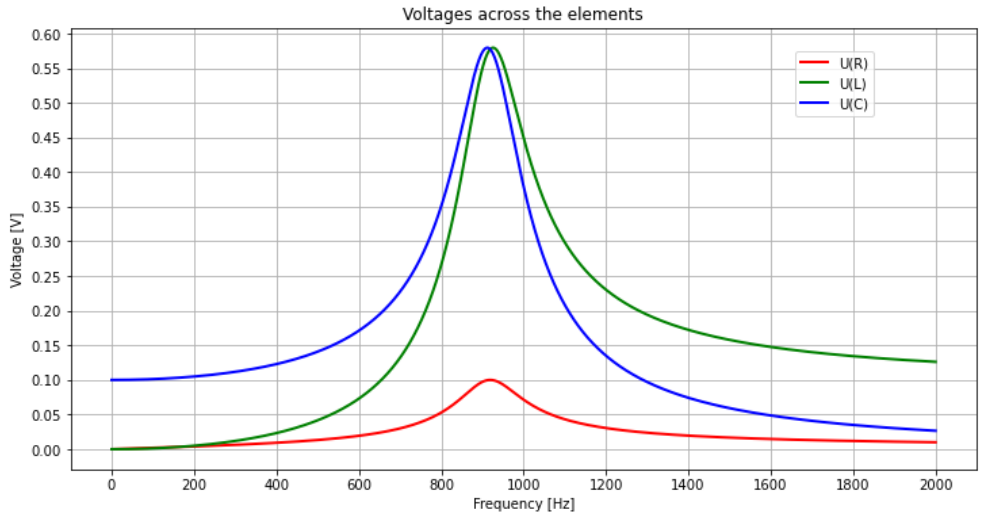


Figure 4. Frequency dependences of the voltages of the three elements of the series RLC circuit with a pronounced voltage resonance

From the graphical results presented, it is found that the voltages on all three elements of the series resonant circuit have a maximum value for the same frequency - this is the resonant frequency of the circuit. In this way, the phenomenon of voltage resonance is formed. When compared with the results presented in [9] it was found that there is a complete match of the frequencies determined with the two different environments. Subsequently, through the obtained data, various calculations can be made regarding the parameters of the resonant circuit, such as determined by its quality factor and bandwidth.

3.2. Scheme of parallel resonant circuit

The schematic of the parallel resonant circuit is shown in Fig.5a. It consists of a sinusoidal voltage source *e(t)*, an inductance, a capacitance and a resistor. And this computational example is taken from a training manual for theoretical electrical engineering students at TU-Sofia [9]. Fig. 5.b shows the LTspise model of the scheme, with the values of the scheme elements and the parameters of the analysis set. In this case, according to the need to study resonance processes, the conduct of AC analysis was also chosen [10].

The scheme and models parameters are:

$e(t) = 10.\sqrt{2}.\sin(3162t - 90°)$, $L_1 = 1\ mH$, $C_1 = 100\ \mu F$, $R_1 = 20\ \Omega$.

The implementation of the LTspice model of a parallel resonant circuit in Python is based on the algorithm presented in Fig.1. The transformation of the model was carried out by means of a developed author's program. Since the model conversion process is identical to the previous case (the same type of analysis is

considered again, only the topology of the circuit diagram is different) the source code of the program for this variant is not presented.

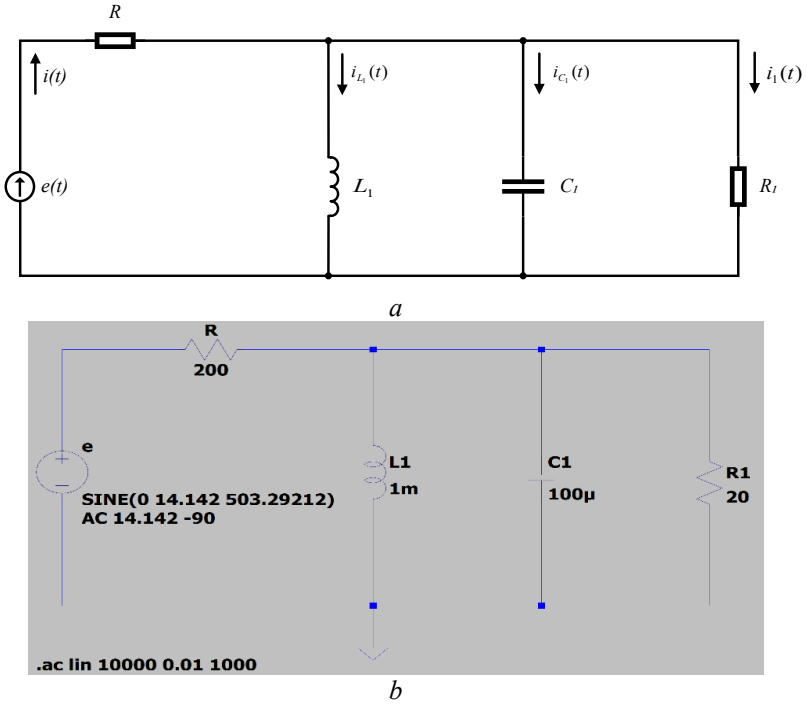


Figure 5. Parallel resonant circuit: a) circuit diagram; b) model in LTspice

In Fig. 6 presents the results of the simulation of the parallel resonant circuit in Python. Specifically, the frequency dependences of the currents through the three elements of the parallel RLC circuit are given.

It can be seen that the currents through all three elements have a maximum value for the same frequency - this is the resonant frequency of the circuit. When compared with the results presented in [9] it was found that there is a complete match of the frequencies determined with the two different environments.

One of the advantages of working with Python is that, based on the data thus obtained, various studies can be made and dependencies between the values of circuit elements, operating modes and basic parameters of the considered electric circuit can be extracted. For example, the dependence of the parameters of the resonant circuit (quality factor and bandwidth) and the values of the elements of the RLC circuit can be experimentally determined. Also to determine the critical resistance value which gives the boundary between resonant and aperiodic mode of operation. This is very convenient for the purposes of learning electronics and electrical engineering, as it immediately enables the verification of theoretical dependencies through computer experiments.

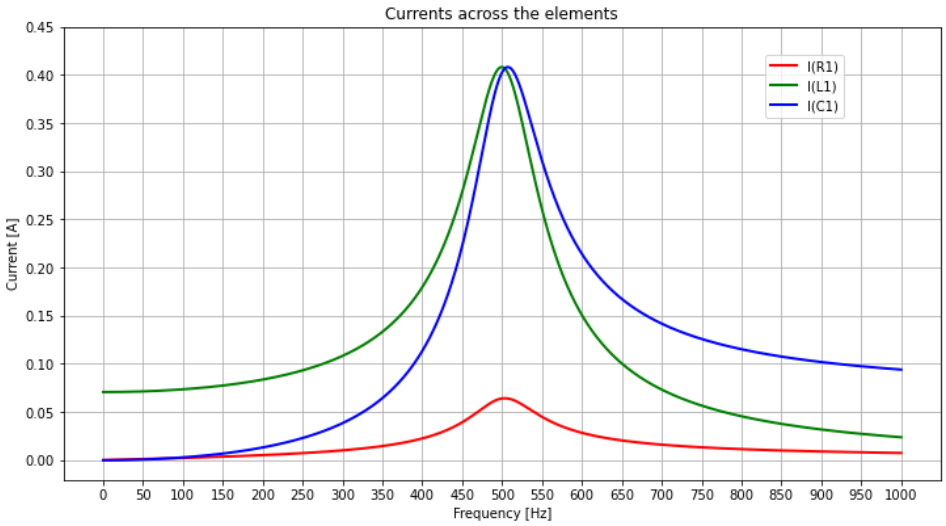


Figure 6. Frequency dependences of the currents through the three elements of the parallel RLC circuit

4. CONCLUSION

The manuscript presents an approach for implementing Spice-based models in Python. An algorithm was developed through which the transformation of the models is carried out. Through two examples of current and voltage resonance research, the method has been validated. The source code of the developed program, which performs the simulation in Python, is also presented. The need for such a transformation is justified due to the fact that as a result of the long work and use of Spice-based computer simulators, many models have been developed, including by companies that manufacture electronic components. On the other hand, Python provides many new possibilities compared to traditional software for modeling electrical and electronic circuits in the field of big data processing, optimization and application of artificial intelligence techniques. In this sense, the presented transformation does not deny the traditional methods of modeling and automated design of electrical and electronic circuits and systems, but offers a development and upgrade of the design and prototyping process, integrating modern information and communication technologies and data science into it. In this way, training in interdisciplinary specialties such as electrical engineering and power electronics is also supported, where knowledge of a complex of diverse fields is necessary such as: mathematics, physics, electrical engineering, circuit engineering, semiconductor elements, control theory. It is a good alternative to the classic methods of studying electrical and electronic circuits. With the use of such innovative approaches, these disciplines become more attractive and appealing to students, and this is very important in view of realizing the green transition and electrification.

ACKNOWLEDGEMENT

This research was carried out within the framework of the project "Artificial Intelligence-Based modeling, design, control and operation of power electronic devices and systems", КП-06-Н57/7/16.11.2021, Bulgarian National Scientific Fund.

REFERENCES

[1] N. T. Hien, L. B. Phuong, V. E. Bolnokin. A special mathematical model for the study of electrical circuits components by the method of differential inclusions. *International Journal on Information Technologies and Security*, vol.15, no.1, 2023, pp. 69-76. https://doi.org/10.59035/GIRQ4188

[2] R. Romansky. Empirical evaluation of the transfer of information resources in active learning. *International Journal on Information Technologies and Security*, vol.15, no.1, 2023, pp. 39-48. https://doi.org/10.59035/WZFU1905.

[3] Marcelo Godoy Simes, Felix A. Farret. Introduction to electrical engineering simulation. in *Modeling Power Electronics and Interfacing Energy Conversion Systems*, IEEE, 2017, pp.1-25, doi: 10.1002/9781119058458.ch1.

[4] S. Weber, C. Duarte. Yield analysis for electrical circuit designs: Many problems and some recent developments in electronic engineering. *IEEE Solid-State Circuits Magazine*, vol. 12, no. 1, pp. 39-52, Winter 2020, doi: 10.1109/MSSC.2019.2939341.

[5] Rashid, Muhammad H., M. H. Rashid. *SPICE for circuits and electronics using PSPICE*. Upper Saddle River, NJ: Prentice Hall, 1995.

[6] K. Kundert, "Life After SPICE," in IEEE Solid-State Circuits Magazine, vol. 3, no. 2, pp. 23-26, Spring 2011, doi: 10.1109/MSSC.2011.942104..

[7] Pine, David J. *Introduction to Python for science and engineering*. CRC press, 2019.

[8] Bittmann, Felix. *Python 3 for Science and Engineering Applications*. Elektor International Media BV, 2020.

[9] Mladenov, V. M., Iatcheva, I. I., Chervenkov, A. G., Petrakieva, S. K., Tsenov, G. T., Petkova, N. S., Chobanov, V. Y., Trushev, I. M., Kirilov, S. M. *Guide for solving problems in Electrical Engineering with LTspice*, TU - Sofia, vol. 1, pp. 140, 2022, Bulgaria, Sofia, Publishing House of the Technical University of Sofia, ISBN 978-619-167-493-0

[10] Asadi, Farzin. *Essential circuit analysis using LTspice*®. Springer International Publishing AG, 2022

Information about the authors:

Stoyan Popov – Assistant in Department of mathematical modeling and numerical methods, Faculty of Applied Mathematics and Informatics, Technical University of Sofia, Sofia, Bulgaria. Current research interests: mathematical and software models, optimal design, innovation methods of design of electric devices.

Nikolay Lyuboslavov Hinov – Associated Professor in Department of Power Electronics, Faculty of Electronic Engineering and Technologies, Technical University of Sofia, Bulgaria. Current research interests: development and design of power electronic converters with application in industrial technologies, electric vehicles, decentralized generation of electricity and energy storage.

Manuscript received on 15 July 2023