P441/P442 - Open Lab Experiment

NON-LINEAR DYNAMICS CIRCUIT

Submitted By

ASHMITA PANDA

ROLL NO. 1811042

School of Physical Sciences

National Institute of Science, Education and Research (NISER), Bhubaneswar

Date of Submission : 24$^{\text{th}}$ September, 2021

Under the Guidance of

Dr. Pratap Kumar Sahoo

Associate Professor

School of Physical Sciences

National Institute of Science Education and Research (NISER), Bhubaneswar

Contents

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

I. Introduction

Chua circuit is the simplest electronic circuit which exhibits the phenomenon of chaos. It was invented by Leon Chua in 1983.

A dynamical system is said to have chaotic behaviour when despite its deterministic nature, it is not predictable. The apparent random behaviour of the system is usually governed by deterministic laws that are highly sensitive to initial conditions. A small change in initial conditions can result in widely varying results.

To exhibit chaos, a circuit must have been :

i.) at least one locally active resistor

ii.) at least one non-linear element

iii.) at least three energy storage units

II. Theoretical Design

2.1 Circuit Elements and Constraints

In order to physically exhibit the phenomenon of chaos Chua decided to design a physical circuit with 3 unstable equilibrium points with further contraints that number of passive elements should be as few as possible and there should be only one non-linear resistor with two terminals which has piecewise linear characteristic.

There must be 3 energy storage elements as the dynamical system must have at least order 3 to be chaotic. He also decided to have only one passive element in the circuit - a linear resistor.
Passive elements are the circuit elements which donot generate power but instead store or dissipate it.

Also since we want to observe oscillations, we cannot have only capacitors or only inductors as all 3 energy storage elements. There must be some combination of both. Chua preferred the combination of two capacitors and one inductor to make the circuit more cost-efficient.

2.2 Possible Configurations

With these constraints in place, there can be 8 possible configurations.

Fig 1. Possible configurations for the circuit

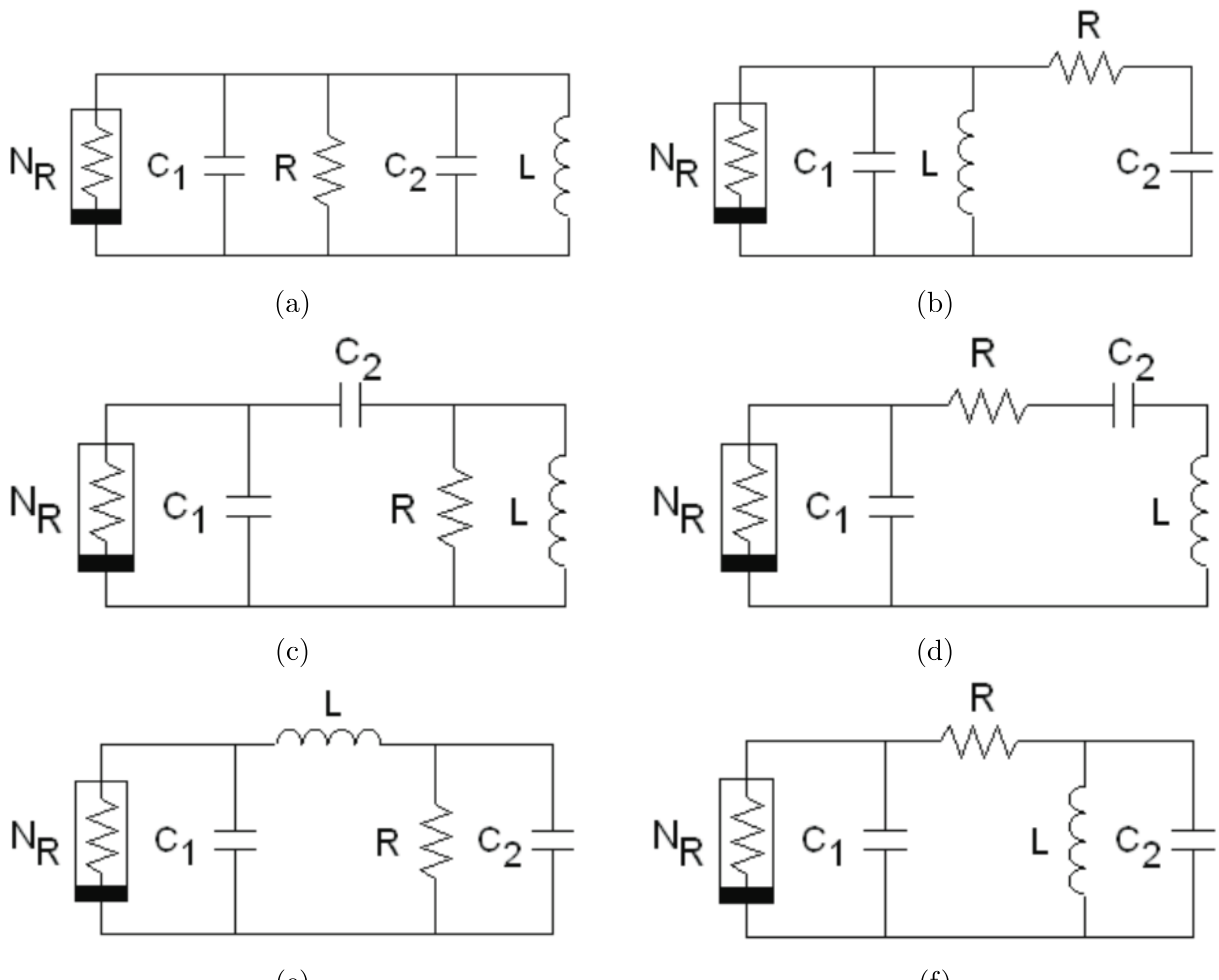
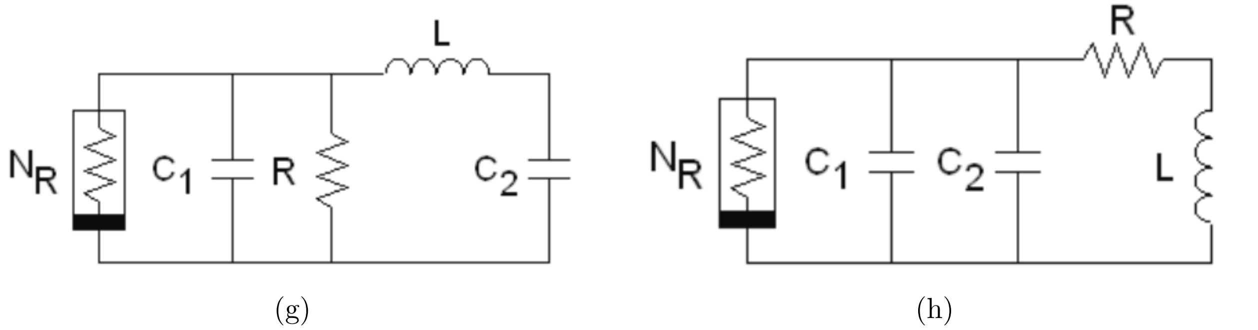


Fig 1. Possible configurations for circuit



Configuration (g) and (h) can be immediately rejected.
In (g) the characteristic of resistance R can be absorbed in the characteristics of non-linear resistor N_R. In (h) the C_1 and C_2 capacitances can be replaced by a single effective capacitor $C = C_1 + C_2$. So in both of these configurations all circuit elements donot give unique contribution. Thus they can be rejected.

For (a) and (b), the DC equilibrium calculations show that non-linear resistor gets short-circuited by the inductor. For (c) and (d), the DC equilibrium calculations show that non-linear resistor terminals are open. So all the four configurations can be rejected.

The remaining configuration (e) and (f) are both valid, but Chua selected configuration (f) because the RLC subcircuit generates oscillations.

2.3 Final Circuit

The final Chua circuit is given as follows :

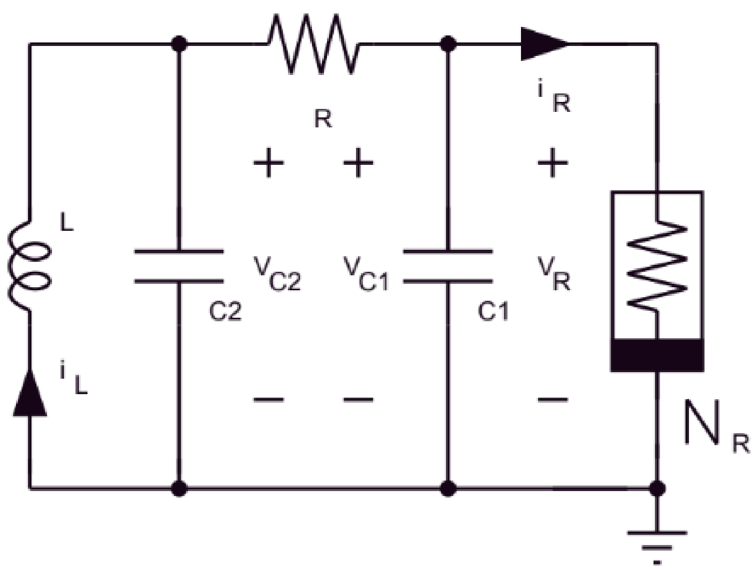


Fig 2. Chua's Circuit

III. State Equations and Simulations

3.1 State Equations

The equations of Chua's circuit are given as a system of three coupled differential equations :

$$C_1 \frac{dv_{C_1}}{dt} = G\left(v_{C_2} - v_{C_1}\right) - g(v_{C_1}) \tag{1}$$

$$C_2 \frac{dv_{C_2}}{dt} = G\left(v_{C_1} - v_{C_2}\right) - i_L \tag{2}$$

$$L \frac{i_L}{dt} = -v_{C_2} \tag{3}$$

where, $G = \dfrac{1}{R}$ is the conductance, and $g(x)$ is a piece-wise linear function. It is given as :

$$g(x) = m_0 x + \frac{1}{2}(m_1 - m_0)\left[|x + B_p| - |x - B_p|\right] \tag{4}$$

where,

$$\begin{aligned} m_0 &\implies \text{slope of outer region} \\ m_1 &\implies \text{slope of inner region} \\ B_P &\implies \text{breakpoints (both positive and negative values)} \end{aligned}$$

3.2 Simulation

The variables were redefined and all constants were taken to right hand side to make handling the equations easier.

$$\frac{dx}{dt} = \frac{1}{C_1}\left\{G\left(y - x\right) - g(x)\right\} \tag{5}$$

$$\frac{dy}{dt} = \frac{1}{C_2}\left\{G\left(x - y\right) - z\right\} \tag{6}$$

$$\frac{dz}{dt} = -\frac{y}{L} \tag{7}$$

where,

$$x \equiv v_{C_1} \quad y \equiv v_{C_2} \quad z \equiv i_L$$

The equation $g(x)$ remains the same as in (4).
The equations are solved numerically using Runge Kutta 4 method in Python. All plots are made using Gnuplot.

3.2.1 Python Codes

The code for RK4 is as follows :

```
#Chua circuit simulations

import math
import handling_files
import numpy as np

#RK4 to solve Chua circuit equations (a system of 3-ODEs)
def RK4_chua(F,b,t,h,N,name):
    handling_files.append_file(name, f'{t} {b[0]} {b[1]} {b[2]}\n')
    for i in range(N):
        K1=F(t,b)
        #
        K2=F(t+h/2, b+np.multiply(K1,h/2))
        #
        K3=F(t+h/2, b+np.multiply(K2,h/2))
        #
        K4=F(t+h, b+np.multiply(K3,h))
        #
        b=b+np.multiply((K1+np.multiply(K2,2)+np.multiply(K3,2)+K4), h/6)
        t=t+h
        handling_files.append_file(name, f'{t} {b[0]} {b[1]} {b[2]}\n')
        #
    return(1)
```

The code inputs the three differential equations as a column vector F which is a function of x, y, z and t (time). x, y and z are arranged as column vector b. For the first iterations, it has the initial values. h is the increment factor. N is the number of iterations. t_0 is the initial time value.

The 'append.file()' function saves the data points (t, x, y and z) after each iteration in a file (filename provided to function as variable 'name'). All codes for manipulation with files is as follows :

```
#Library for handling files and their contents

# READ FILE
def read_matrix(x): #more than one column #parameter = name of file
    f=open(x,'r') #'r' ==> read only
    X=[[float(num) for num in line.split('\t')] for line in f]
    f.close()
    return(X)

#
    ###########################################################################
```

```python

# APPEND FILE
def append_file(x, str): #arguments = name of file, string to append
    f=open(x, 'a') #'a' ==> append file
    f.write(str)
    f.close()
    #

# ##########################################################################

#WRITE AT BEGINNING (hopefully)
def write_beginning(x, str):
    f=open(x, 'r+')
    old=f.read()
    f.seek(0)
    f.write(str + old)
    f.close()
    # it works :D

# PRINT CONTENTS OF A TEXT FILE
def print_file(x): #argument = name of file
    f=open(x, 'r')
    contents=f.read()
    print(contents)

# ###########################################################################
```

3.2.2 Plots with Dimensionless Constants

The following values were used for the constants :

$$G = 0.7 \quad C_1 = 1/9 \quad C_2 = 1 \quad L = 1/7 \quad B_p = 1 \quad m_0 = -0.5 \quad m_1 = -0.8$$

The code for defining the function, initial values and calling the function is :

```python
# Dimensionless Chua Circuit

import math
import sys
import numpy as np

sys.path.append('/home/ashmita/Desktop/ASHMITA/APanda_Lib')
```

```python
# importing all files at once, now we just need to write function name to access it
from APanda_Lib import *

import chua_circuit_simulations

#dimensionless chua
R=1.4285 # R corresponding to G=0.7
C1=1/9
C2=1
L=1/7
Bp=1
m0=-0.5
m1=-0.8

x0=0.1
y0=0.0
z0=0.0
t0=0.0

b0=[x0, y0, z0]

def Yfunc(t,b):
    x,y,z=b
    gx=m0*x+0.5*(m1-m0)*(abs(x+1)-abs(x-1))
    Y=[(1/C1)*((1/R)*(y-x)-gx), (1/C2)*((1/R)*(x-y)+z), (-1/L)*y]
    return Y
    #

h=0.1
N=5000
path="/home/ashmita/Desktop/ASHMITA/NISER Study/7th Semester/Open Lab/Non-Linear Circuit/Dimensionless/"
name=f'dimensionless'
n=path+name
f=open(n, "w")
f.close()
out2=chua_circuit_simulations.RK4_chua(Yfunc,b0,t0,h,N,n)
#
```