

程序代码

1、相图模拟代码

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint

# constants
global C1, C2, L, R, G
C1 = 11.061e-9 # F
C2 = 101.0e-9 # F
L = 23.7095e-3 # H
R_start = 1700 #omega
R_end = 2069 #omega
dR = 1

# initial condition
t0 = 0
y0 = np.array([0.0,0.0,0.001]) # [U1, U2, I]
t_final = 0.1
t_total = 20000

# I-V curve of the nonlinear negative resistance
def g(U):
    Ga = -0.00076 # omega-1 uncertainty: 0.1*10-4
    Gb = -0.000409 # omega-1 uncertainty: 0.06*10-4
    E = 15.0 #V
    g = Gb*U+(Gb-Ga)/2*(abs(U-E)-abs(U+E))
    return g

# chua's circuit
def f(y,t):
    y1 = (G*(y[1]-y[0])-g(y[0]))/C1
    y2 = (G*(y[0]-y[1])+y[2])/C2
    y3 = -y[1]/L
    return np.array([y1, y2, y3])

time=np.linspace(t0,t_final,t_total)
R_list = [R_start + i*dR for i in range(int((R_end-R_start)/dR))]
for R in R_list:
    G = 1/R
    y = odeint(f,y0,time)

    # phase plot
    plt.plot(y[4000:,0],y[4000:,1] , label = 'R = %f omega'%R )
    plt.title('Phase Plot')
    plt.xlabel('U1/V')
    plt.ylabel('U2/V')
    plt.legend()
    plt.savefig('%f omega.jpg'%R)
    plt.clf()
    print('%f omega Done!'%R)
```

2、U-t图及数据模拟代码

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
import pandas as pd

# constants
global C1, C2, L, R,G
C1 = 11.061e-9 # F
C2 = 101.0e-9 # F
L = 23.7095e-3 # H
R_start = 1710 # omega
R_end = 2065 # omega
dR = 1

# initial condition
t0 = 0
y0 = np.array([0.0,0.0,0.001]) # [U1, U2, I]
t_final = 0.1
t_total = 20000

# I-V curve of the nonlinear negative resistance
def g(U):
    Ga = -0.00076 # omega-1 uncertainty: 0.1*10-4
    Gb = -0.000409 # omega-1 uncertainty: 0.06*10-4
    E = 15.0 #V
    g = Gb*U+(Gb-Ga)/2*(abs(U-E)-abs(U+E))
    return g

# chua's circuit
def f(y,t):
    y1 = (G*(y[1]-y[0])-g(y[0]))/C1
    y2 = (G*(y[0]-y[1])+y[2])/C2
    y3 = -y[1]/L
    return np.array([y1, y2, y3])

time=np.linspace(t0,t_final,t_total)
R_list = [R_start + i*dR for i in range(int((R_end-R_start)/dR))]
for R in R_list:
    G = 1/R
    y = odeint(f,y0,time)

    # simulation data plot
    plt.plot(time[4000:6000:],y[4000:6000,0],label = 'U1') #eliminate unstable
solutions
    plt.plot(time[4000:6000:], y[4000:6000,1], label='U2')
    plt.xlabel('t')
    plt.ylabel('U')
    plt.title('R = %f omega'% round(R, 1))
    plt.legend()
    plt.savefig('%f omega.jpg' % round(R, 1))
    plt.clf()
    print('%f omega Done!' % round(R, 1))

# output simulating data
```

```
dataframe = pd.DataFrame({'t': time[4000:].tolist(), 'U1':  
y[4000:,0].tolist(), 'U2': y[4000:,1].tolist()})  
dataframe.to_csv("%f_omega.csv"%round(R,1), index=False, sep=',')
```

3、FFT分析代码

实验数据分析代码如下所示：

```
import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd  
  
# load data  
filename = '20000101_022739s.csv'  
# '20000101_021718dd.csv' 双吸引子  
# '20000101_021921ss.csv' 单吸引子  
# '20000101_022112tr.csv' 三周期  
# '20000101_022321zhenfa.csv' 阵发混沌  
# '20000101_022614f.csv' 四周期  
# '20000101_022700d.csv' 双周期  
# '20000101_022739s.csv' 单周期  
CH2_raw = pd.read_csv(filename, sep=',', header=9, usecols=[2])  
CH1_raw = pd.read_csv(filename, sep=',', header=9, usecols=[1])  
array1 = CH1_raw.values  
array2 = CH2_raw.values  
CH1 = []  
CH2 = []  
for i in range(len(array1)):  
    CH1.append(float(array1[i,0]))  
    CH2.append(float(array2[i, 0]))  
t = np.arange(len(CH1))  
  
# FFT  
CH1_fft = np.fft.rfft(CH1)  
power1 = [abs(c) for c in CH1_fft]  
CH2_fft = np.fft.rfft(CH2)  
power2 = [abs(c) for c in CH2_fft]  
  
# plot the power spectrum  
loc = t.tolist().index(1000)  
plt.plot(t[:loc], power1[:loc])  
plt.xlabel('Frequency')  
plt.ylabel('Power')  
plt.title('U1 Power Spectrum')  
plt.show()  
  
plt.plot(t[:loc], power2[:loc])  
plt.xlabel('Frequency')  
plt.ylabel('Power')  
plt.title('U2 Power Spectrum')  
plt.show()
```

模拟数据分析代码在读取文件数据时参数有一些差别，如下所示，其余都与上面分析实验数据的代码一致。

```
CH2_raw = pd.read_csv(filename, sep=',', header=1, usecols=[2])
CH1_raw = pd.read_csv(filename, sep=',', header=1, usecols=[1])
```

4、分岔图代码

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
import pandas as pd

# constants
global C1, C2, L, R, G
C1 = 11.061e-9 # F
C2 = 101.0e-9 # F
L = 23.7095e-3 # H
R_start = 1980 #omega
R_end = 2066 #omega
dR = 0.01

# initial condition
t0 = 0
y0 = np.array([0.0, 0.0, 0.001]) # [U1, U2, I]
t_final = 0.1
t_total = 20000

# I-V curve of the nonlinear negative resistance
def g(U):
    Ga = -0.00076 # omega^(-1) uncertainty: 0.1*10^(-4)
    Gb = -0.000409 # omega^(-1) uncertainty: 0.06*10^(-4)
    E = 15.0 #V
    g = Gb*U+(Gb-Ga)/2*(abs(U-E)-abs(U+E))
    return g

# chua's circuit
def f(y, t):
    y1 = (G*(y[1]-y[0])-g(y[0]))/C1
    y2 = (G*(y[0]-y[1])+y[2])/C2
    y3 = -y[1]/L
    return np.array([y1, y2, y3])

# find local minimum function
def mini(data):
    '''input the data list'''
    length = len(data)
    ans = []
    for i in range(2, length-2):
        if data[i]<data[i-1] and data[i]<data[i-2] and data[i]<data[i+1] and
data[i]<data[i+2]:
            ans.append(data[i])
    return ans

time=np.linspace(t0,t_final,t_total)
R_list = [R_start + i*dR for i in range(int((R_end-R_start)/dR))]

R1list = [] # for bifurcation diagram
R2list = []
```

```

U1min = [] # also for bifurcation diagram
U2min = []

for R in R_list:
    G = 1/R
    y = odeint(f,y0,time)

    # find the local minimum
    y1list= y[4000:,0].tolist()
    U1min += mini(y1list)
    R1list += [R+i-i for i in range(len(mini(y1list)))]
    y2list = y[4000:, 1].tolist()
    U2min += mini(y2list)
    R2list += [R + i - i for i in range(len(mini(y2list)))]
    print('R = %f omega'%round(R,3))

# plot the bifurcation diagram
plt.scatter(R1list,U1min,s=0.1,alpha = 1)
plt.xlabel('R/omega')
plt.ylabel('U1min')
plt.title('Bifurcation Diagram for U1min')
plt.savefig('U1min')
plt.clf()

plt.scatter(R2list,U2min,s=0.1,alpha = 1)
plt.xlabel('R/omega')
plt.ylabel('U2min')
plt.title('Bifurcation Diagram for U2min')
plt.savefig('U2min')
plt.clf()

# save data
dataframe1 = pd.DataFrame({'R': R1list, 'U1min': U1min})
dataframe1.to_csv("U1min.csv", index=False, sep=',')
dataframe2 = pd.DataFrame({'R': R2list, 'U1min': U2min})
dataframe2.to_csv("U2min.csv", index=False, sep=',')

```