# Group 3

混沌結合保密通訊

## 先來跑一次程式吧！

## 大綱

- 目的
- 原理
- 程式步驟
- 程式解說
- 成果
- 討論

## 目 的

- 以混沌作為載波傳遞訊息
- 要知道該載波混沌的初始參數才能破解 =>保密通訊

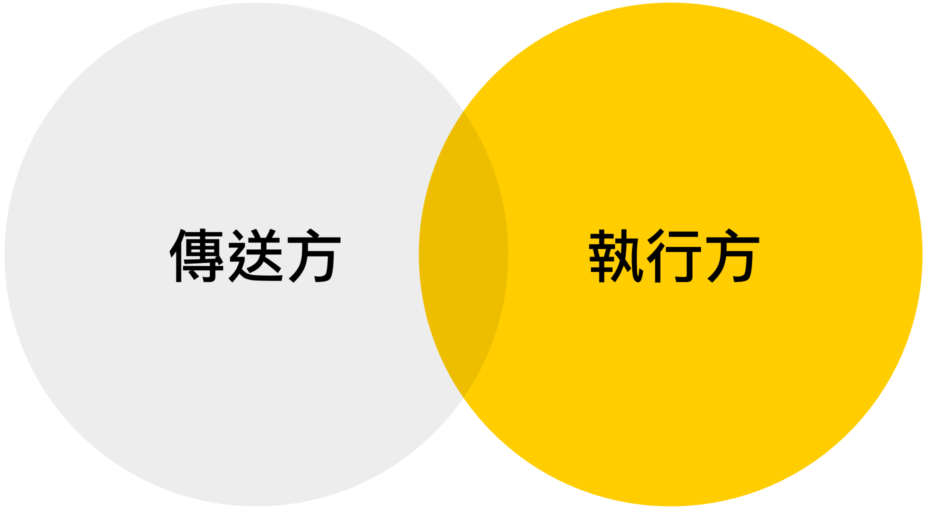
## 原理

- 載波就是攜帶信息/信號的波形。
- 將要傳遞的信號加載到載波的信號上，接收方按照載波的頻率來接收傳遞的信號，有意義的信號波的波幅與無意義的信號的波幅是不同的，將這些信號提取出來就是傳送方要傳遞的信息。

## 原理

- 混沌受初始狀態影響的敏感性，初始條件非常微小的變動也可以導致最終狀態的巨大差別。所以唯有知道初始條件的接收方可以製造出相同的混沌消掉載波，提取出有意義的信號。

## 程式步驟



## 程式步驟

傳送方：

將要傳遞的聲音訊息錄成wav檔

讀取wav檔畫出振幅-時間圖

將聲音訊息分析成雙聲道分別混入3-D的混沌的XY軸

播出混和後的音檔

## 程式步驟

接收方：

接收混合後的訊號與初始參數

根據參數重新製造混沌

將混合訊號扣除混沌

輸出原始訊號並播放

## 程式解說

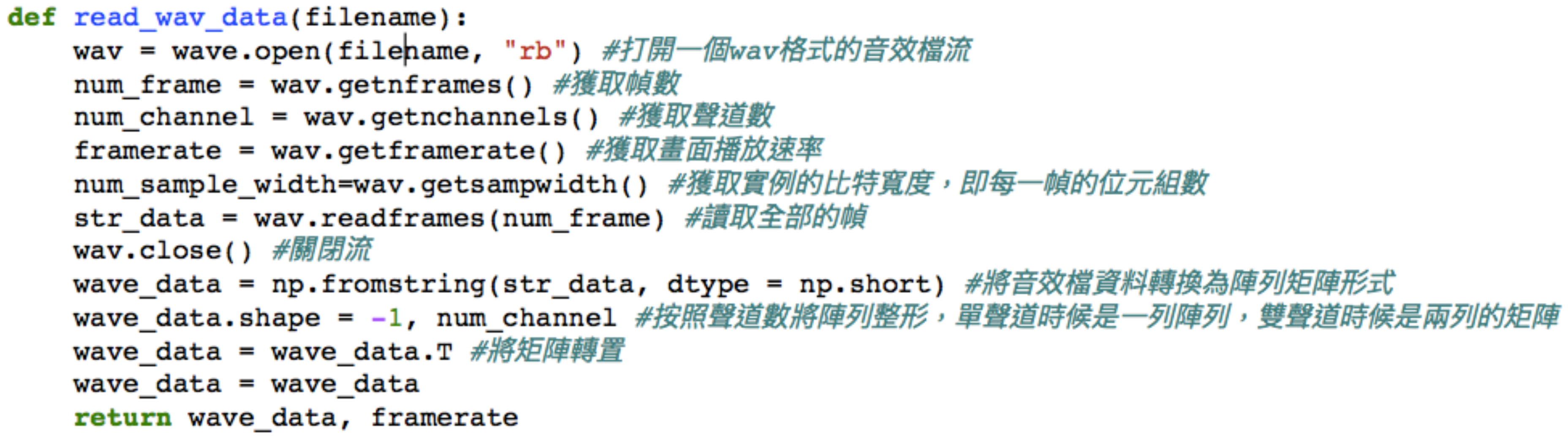
- 匯入套件
- 讀取wav檔並寫入txt檔
- 播放wav檔
- 輸出wav檔
- 混沌製造
- 彈出視窗

## 匯入套件

```
import numpy as np
import scipy.integrate as integrate
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

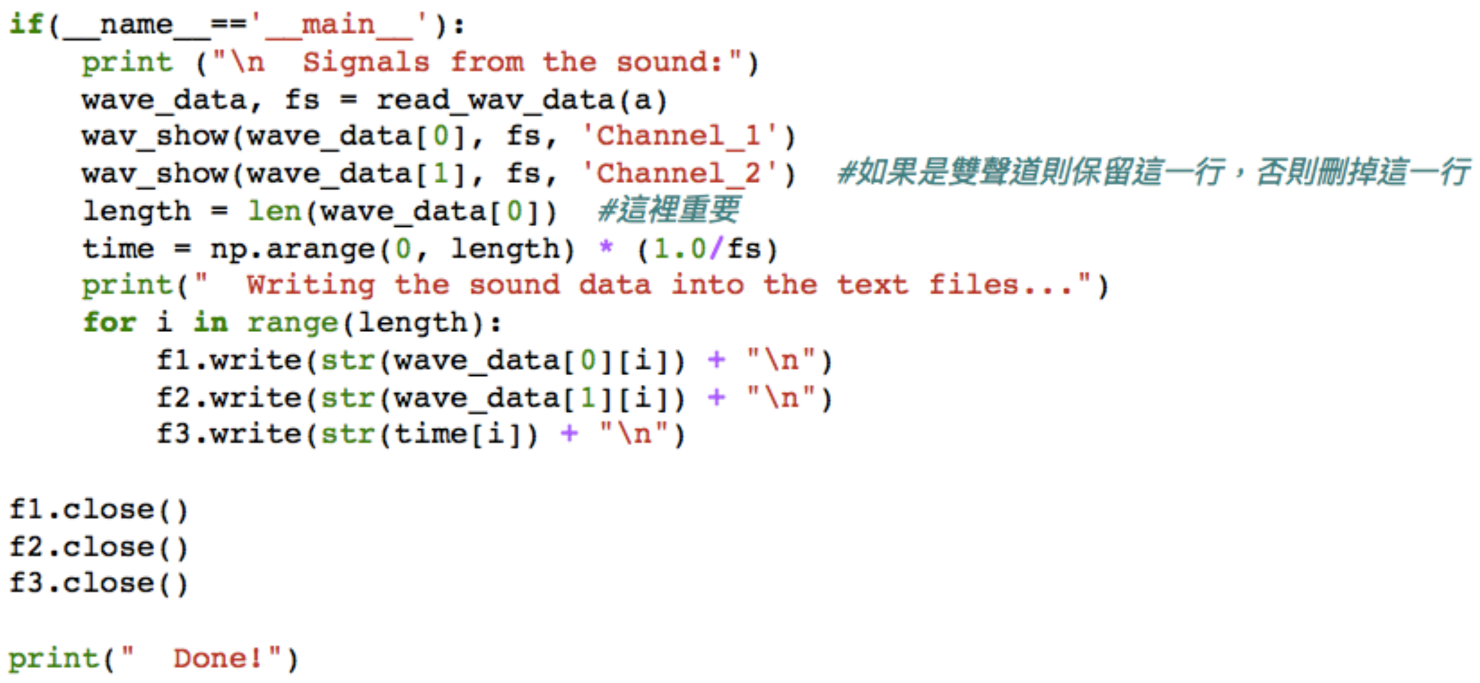
- `import pyaudio`
- `import wave`
- `import tkinter as tk`
- `import tkinter.messagebox`

## 讀取wav檔並寫入txt檔



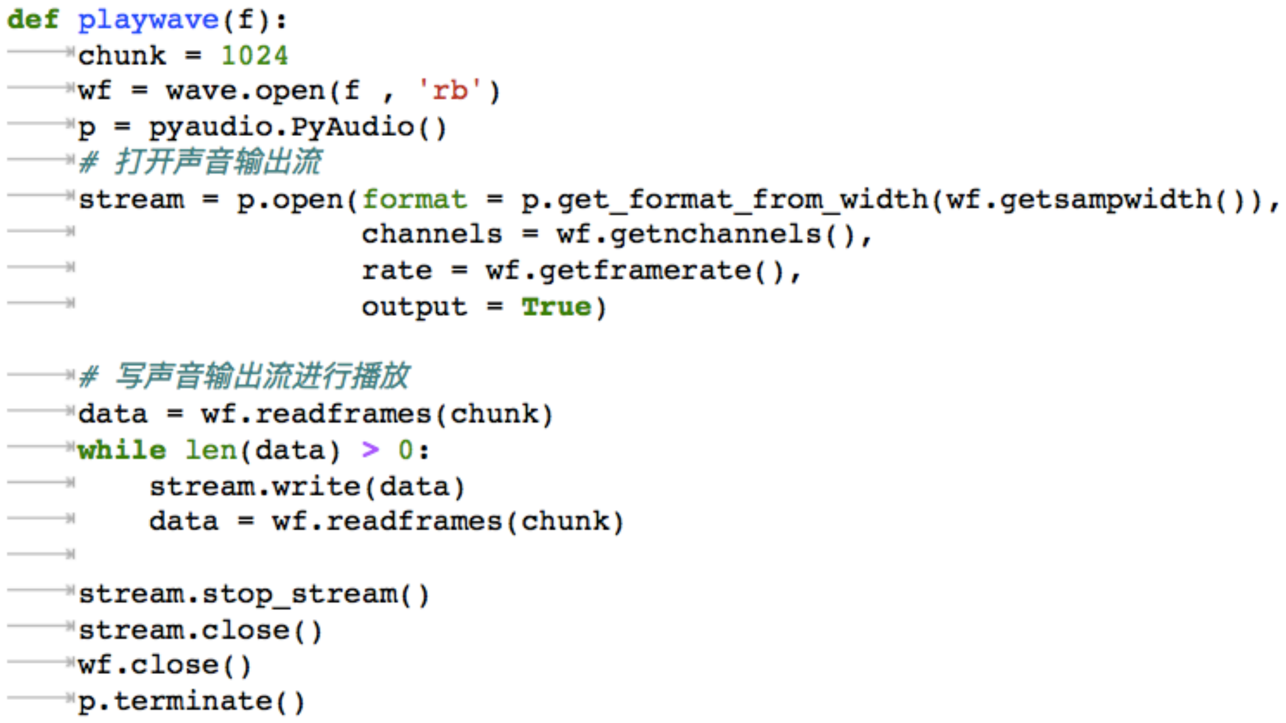
```
def read_wav_data(filename):
    wav = wave.open(filename, "rb") #打開一個wav格式的音效檔流
    num_frame = wav.getnframes() #獲取幀數
    num_channel = wav.getnchannels() #獲取聲道數
    framerate = wav.getframerate() #獲取畫面播放速率
    num_sample_width=wav.getsampwidth() #獲取實例的比特寬度，即每一幀的位元組數
    str_data = wav.readframes(num_frame) #讀取全部的幀
    wav.close() #關閉流
    wave_data = np.fromstring(str_data, dtype = np.short) #將音效檔資料轉換為陣列矩陣形式
    wave_data.shape = -1, num_channel #按照聲道數將陣列整形，單聲道時候是一列陣列，雙聲道時候是兩列的矩陣
    wave_data = wave_data.T #將矩陣轉置
    wave_data = wave_data
    return wave_data, framerate
```

## 讀取wav檔並寫入txt檔



```
if(__name__=='__main__'):
    print ("\n  Signals from the sound:")
    wave_data, fs = read_wav_data(a)
    wav_show(wave_data[0], fs, 'Channel_1')
    wav_show(wave_data[1], fs, 'Channel_2')  #如果是雙聲道則保留這一行，否則刪掉這一行
    length = len(wave_data[0])  #這裡重要
    time = np.arange(0, length) * (1.0/fs)
    print("  Writing the sound data into the text files...")
    for i in range(length):
        f1.write(str(wave_data[0][i]) + "\n")
        f2.write(str(wave_data[1][i]) + "\n")
        f3.write(str(time[i]) + "\n")

f1.close()
f2.close()
f3.close()

print("  Done!")
```

## 播放wav檔



```python
def playwave(f):
    chunk = 1024
    wf = wave.open(f , 'rb')
    p = pyaudio.PyAudio()
    # 打开声音输出流
    stream = p.open(format = p.get_format_from_width(wf.getsampwidth()),
                    channels = wf.getnchannels(),
                    rate = wf.getframerate(),
                    output = True)

    # 写声音输出流进行播放
    data = wf.readframes(chunk)
    while len(data) > 0:
        stream.write(data)
        data = wf.readframes(chunk)

    stream.stop_stream()
    stream.close()
    wf.close()
    p.terminate()
```

## 輸出wav檔

```python
framerate = 44100
time = length / framerate

out1_array = np.array([0.0] * length)
out2_array = np.array([0.0] * length)


for l in range(length):
    out1_array[l] = out1[l]
    out2_array[l] = out2[l]


wave_data1 = out1_array
wave_data1 = wave_data1.astype(np.short)

h1 = wave.open(r"mix1.wav", "wb")  # 打開.wav檔
h1.setnchannels(1)  #配置聲道數、量化位元數和取樣頻率
h1.setsampwidth(2)
h1.setframerate(framerate)
h1.writeframes(wave_data1.tostring())  #將wav_data轉換為二進位資料寫入檔

print("  File 'mix1.wav' created.")

h1.close()
```

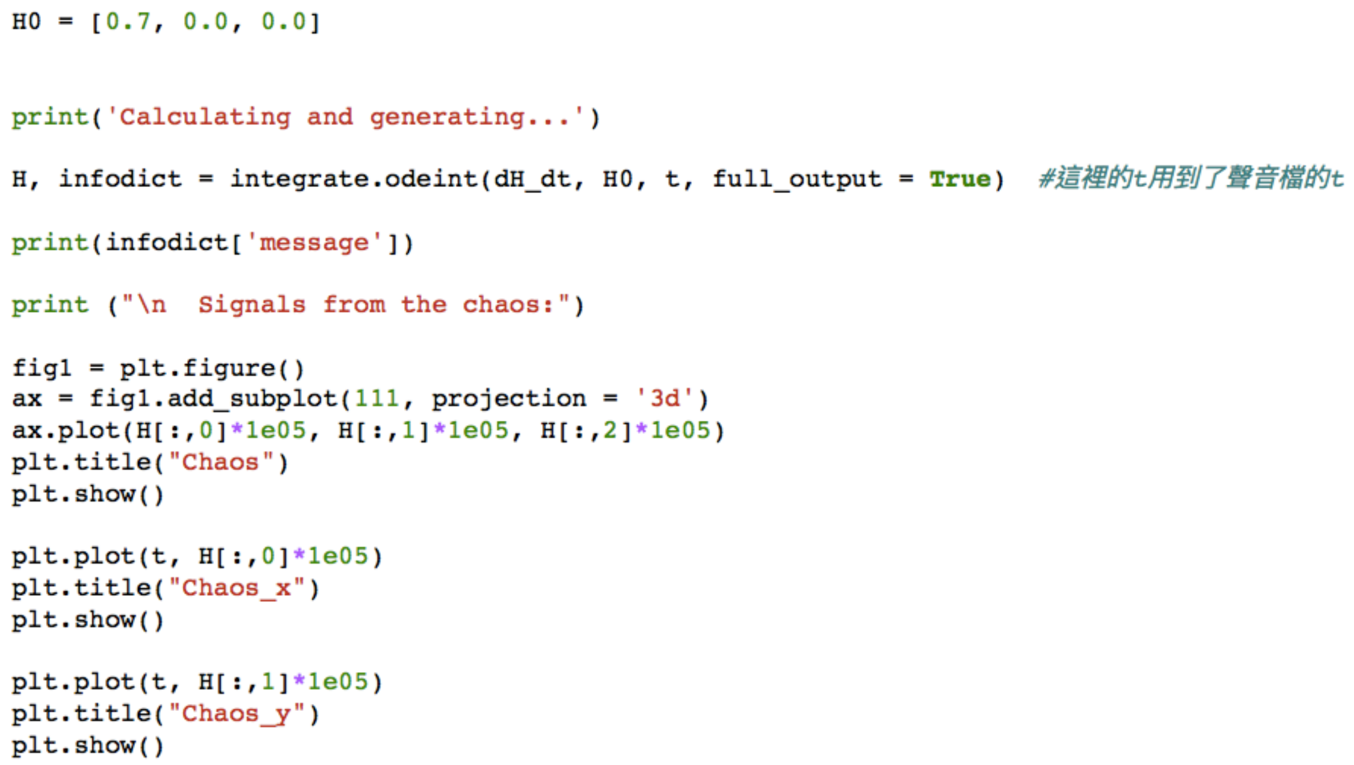
## 製造混沌

```python
ans = input('Use the default parameters to generate the chaos (y/n)? ')
if ans == 'y':
    c0 = 15.6 * 600
    c1 = 1.0 * 600
    c2 = 28.0 * 600
    m0 = -1.143
    m1 = -0.714
if ans == 'n':
    print('Enter the values of the following parameters: ')
    c0 = float(input('c0 = '))
    c1 = float(input('c1 = '))
    c2 = float(input('c2 = '))
    m0 = float(input('m0 = '))
    m1 = float(input('m1 = '))


s1 = open("setting.txt", "w")
s1.write(str(c0)+"\n")
s1.write(str(c1)+"\n")
s1.write(str(c2)+"\n")
s1.write(str(m0)+"\n")
s1.write(str(m1)+"\n")
s1.write(str(length)+"\n")
```

## 製造混沌

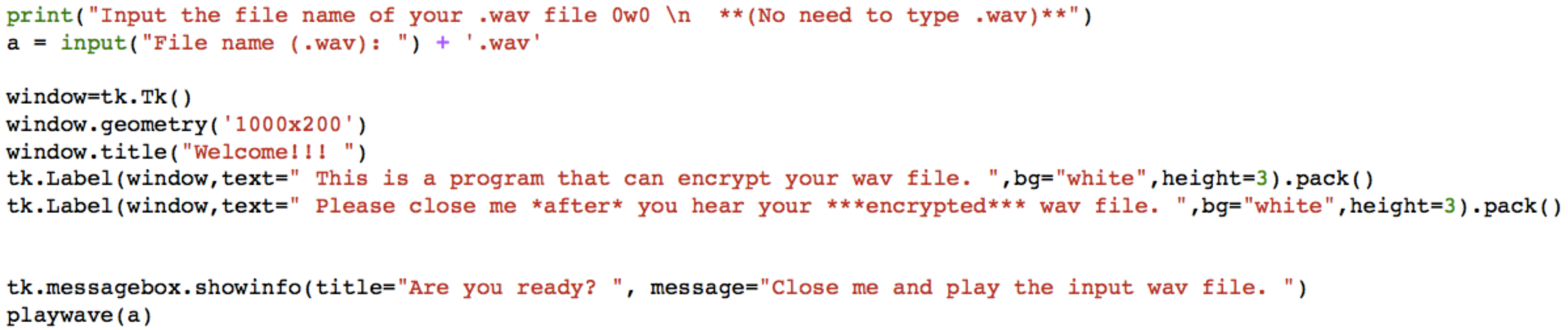
```
def f(x):
    f = m1*x + (m0-m1)/2.0*(abs(x+1.0)-abs(x-1.0))
    return f

def dH_dt(H, t=0):
    return np.array([c0 * (H[1]-H[0]-f(H[0])),
                     c1 * (H[0]-H[1]+H[2]),
                     -c2 * H[1]])
```

## 製造混沌



```
H0 = [0.7, 0.0, 0.0]


print('Calculating and generating...')

H, infodict = integrate.odeint(dH_dt, H0, t, full_output = True)   #這裡的t用到了聲音檔的t

print(infodict['message'])

print ("\n  Signals from the chaos:")

fig1 = plt.figure()
ax = fig1.add_subplot(111, projection = '3d')
ax.plot(H[:,0]*1e05, H[:,1]*1e05, H[:,2]*1e05)
plt.title("Chaos")
plt.show()

plt.plot(t, H[:,0]*1e05)
plt.title("Chaos_x")
plt.show()

plt.plot(t, H[:,1]*1e05)
plt.title("Chaos_y")
plt.show()
```

## 彈出視窗

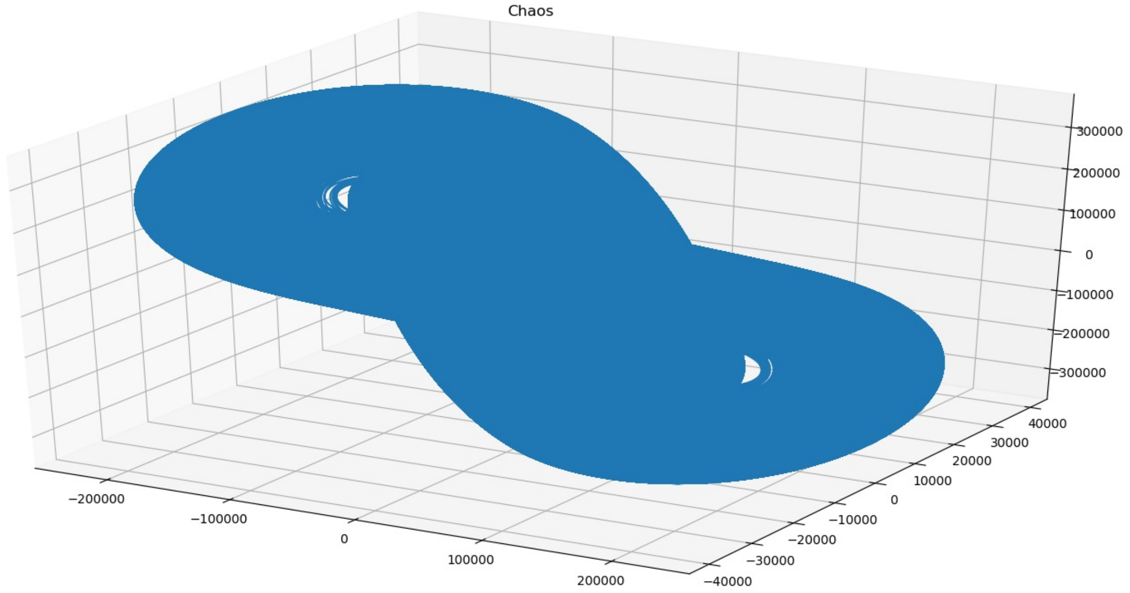


```
print("Input the file name of your .wav file 0w0 \n  **(No need to type .wav)**")
a = input("File name (.wav): ") + '.wav'

window=tk.Tk()
window.geometry('1000x200')
window.title("Welcome!!! ")
tk.Label(window,text=" This is a program that can encrypt your wav file. ",bg="white",height=3).pack()
tk.Label(window,text=" Please close me *after* you hear your ***encrypted*** wav file. ",bg="white",height=3).pack()


tk.messagebox.showinfo(title="Are you ready? ", message="Close me and play the input wav file. ")
playwave(a)
```

## 成果

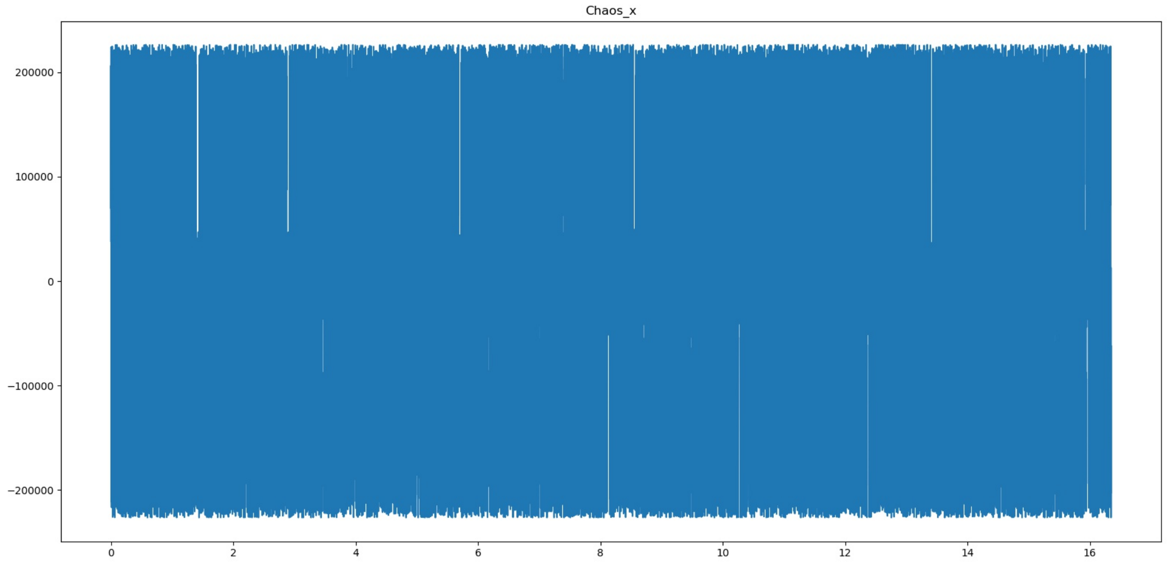
- 混沌波形
- 聲波波形
- 混和波形
- 誤差分析

## 混沌波形

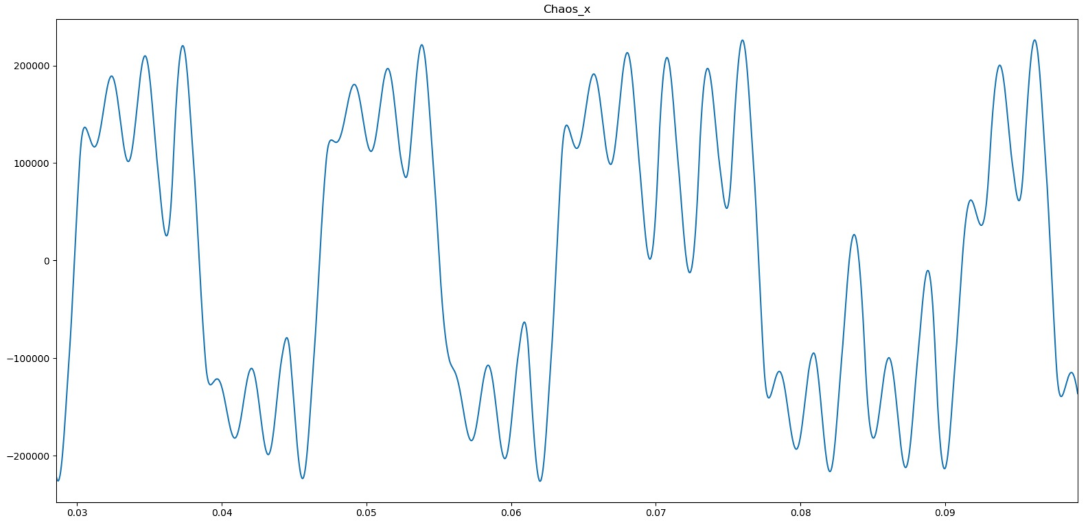


Chaos
300000
200000
100000
0
−100000
−200000
−300000
40000
30000
20000
10000
0
−10000
−20000
−30000
−40000
−200000
−100000
0
100000
200000

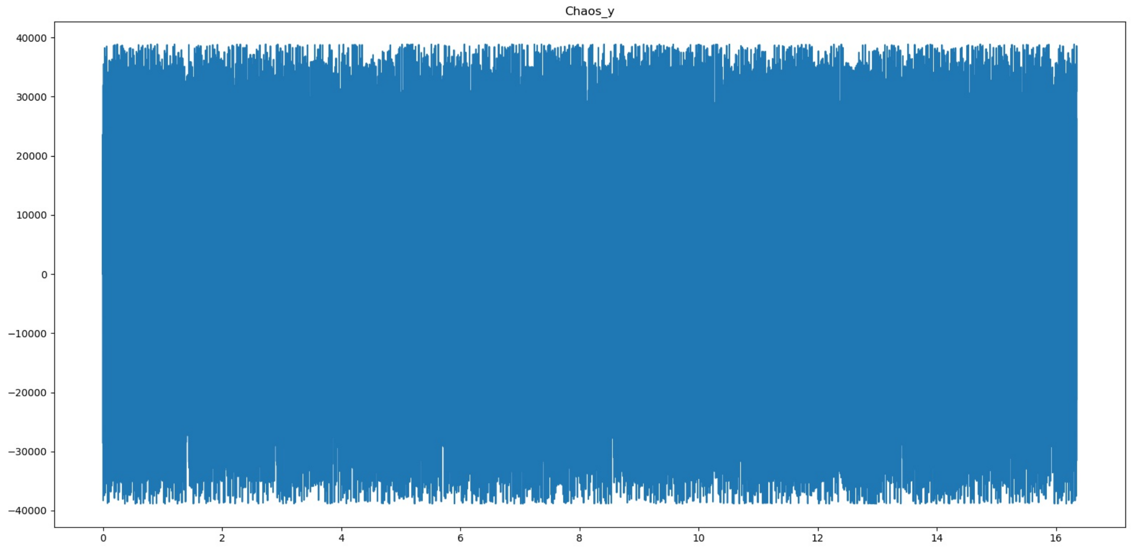
## 混沌波形



## 混沌波形

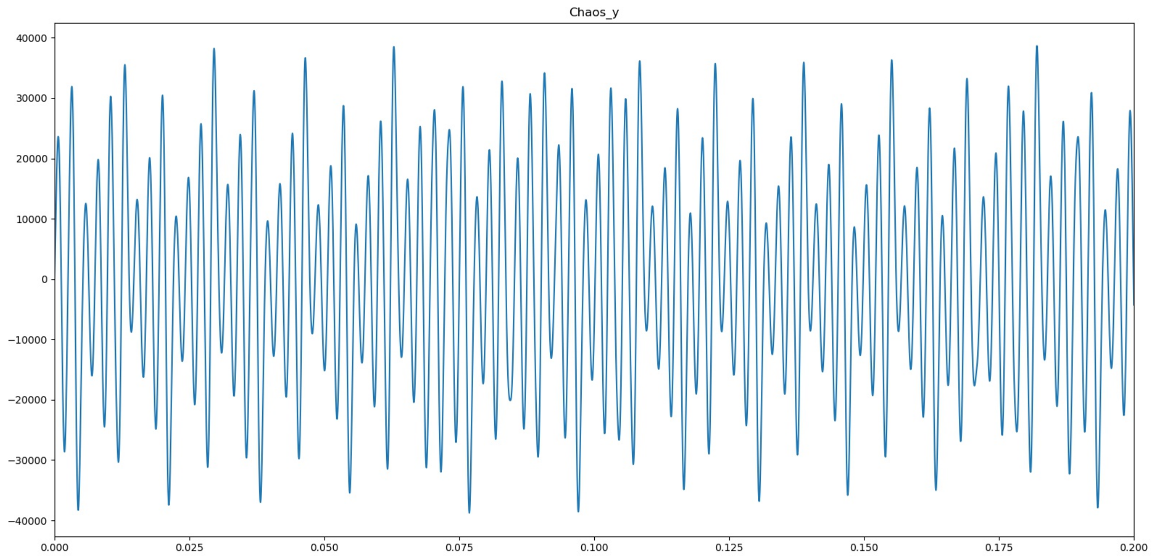

Chaos_x
200000
100000
0
−100000
−200000
0.03
0.04
0.05
0.06
0.07
0.08
0.09

## 混沌波形

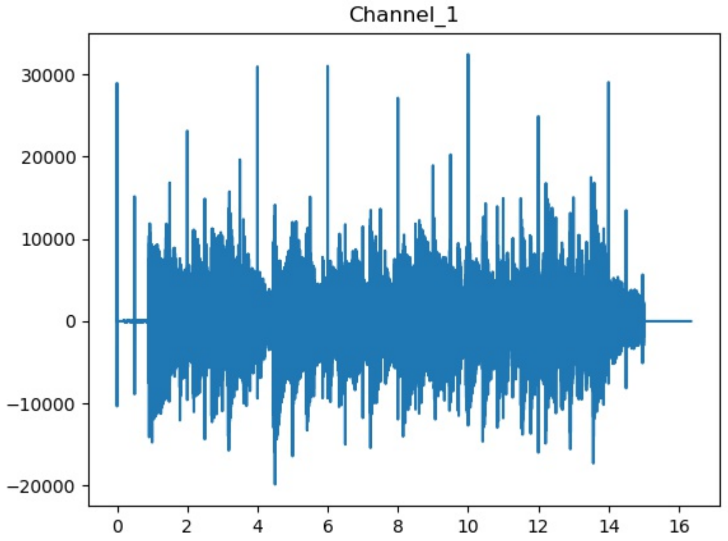


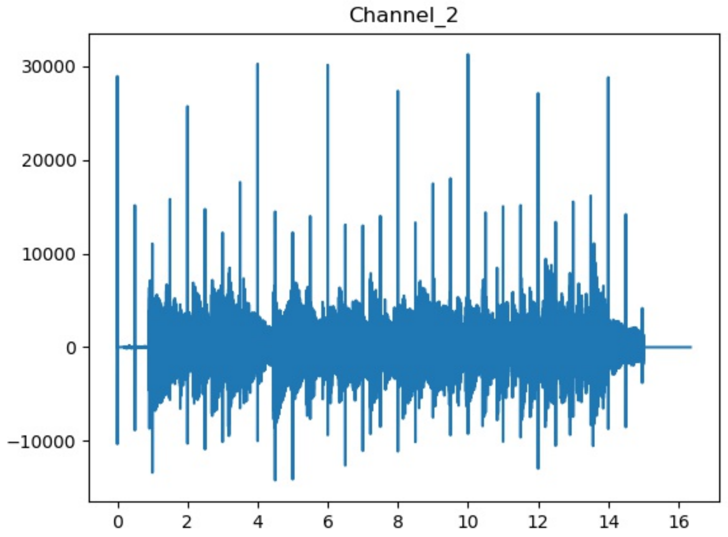
Chaos_y
40000
30000
20000
10000
0
−10000
−20000
−30000
−40000
0
2
4
6
8
10
12
14
16

## 混沌波形

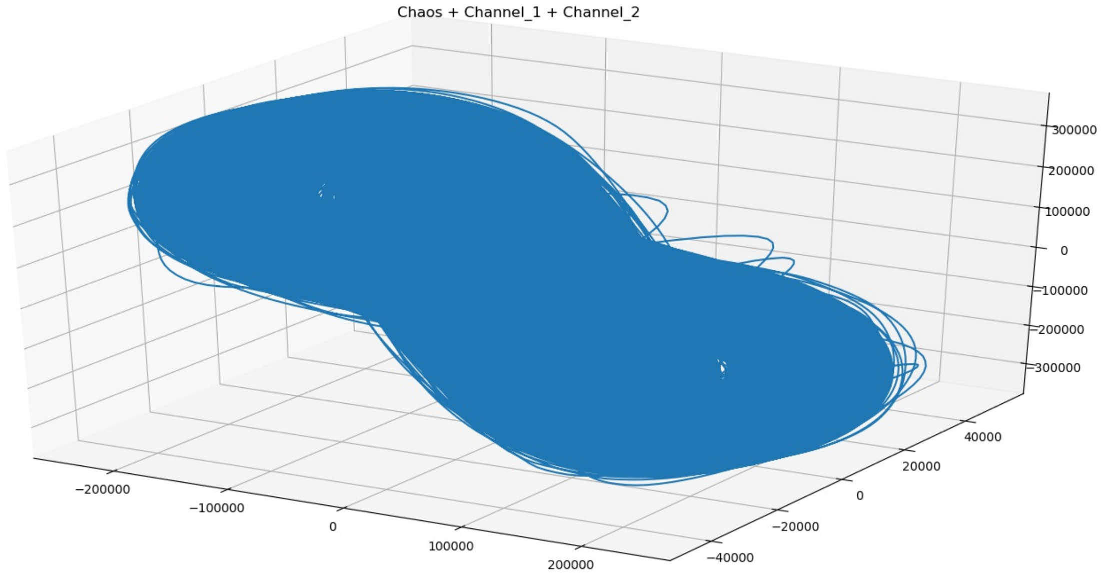

Chaos_y
40000
30000
20000
10000
0
−10000
−20000
−30000
−40000
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200

## 聲波波形


Channel_1
30000
20000
10000
0
−10000
−20000
0
2
4
6
8
10
12
14
16

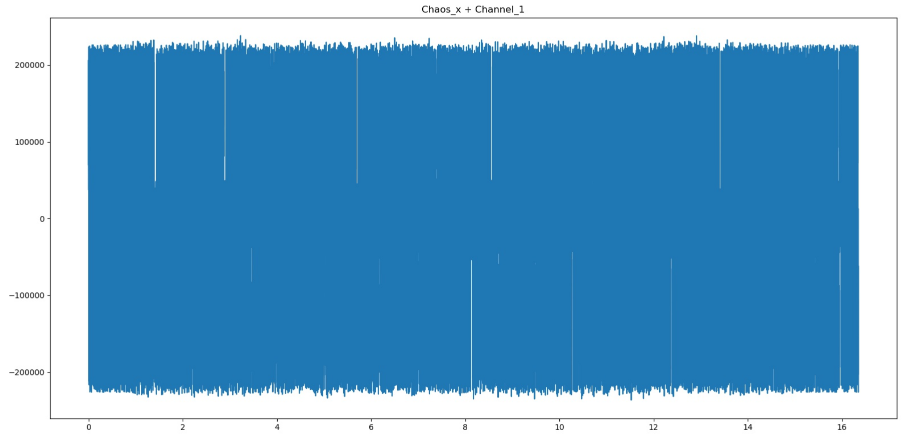

Channel_2
30000
20000
10000
0
−10000
0
2
4
6
8
10
12
14
16

## 混合波形

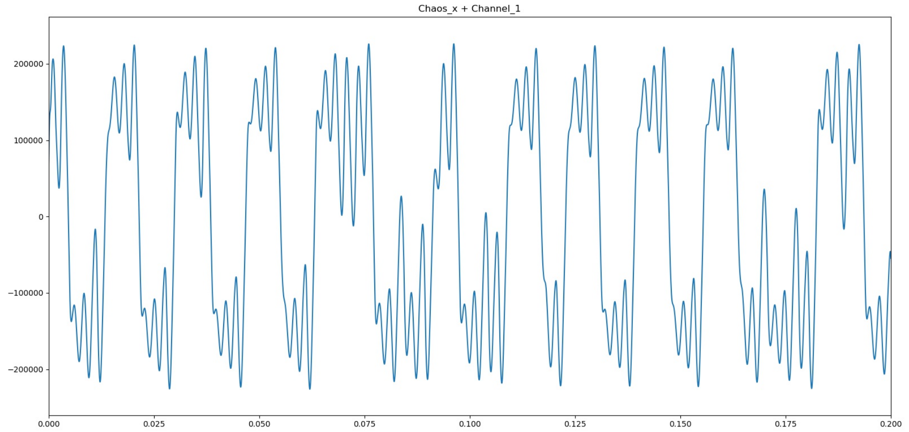


Chaos + Channel_1 + Channel_2
300000
200000
100000
0
−100000
−200000
−300000
−200000
−100000
0
100000
200000
40000
20000
0
−20000
−40000

## 混合波形

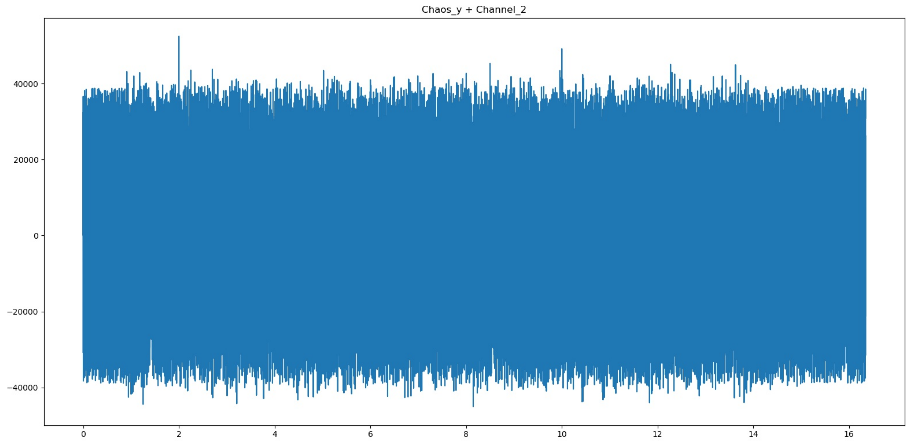


Chaos_x + Channel_1
200000
100000
0
−100000
−200000
0
2
4
6
8
10
12
14
16

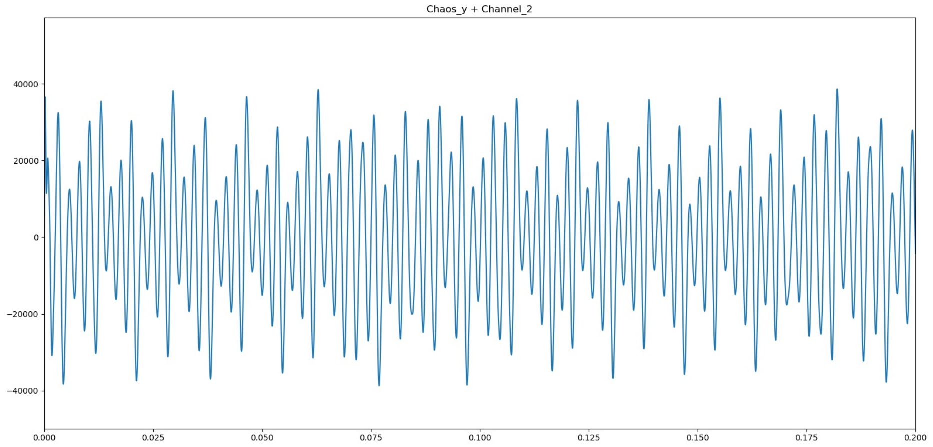


Chaos_x + Channel_1
200000
100000
0
−100000
−200000
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200

## 混合波形

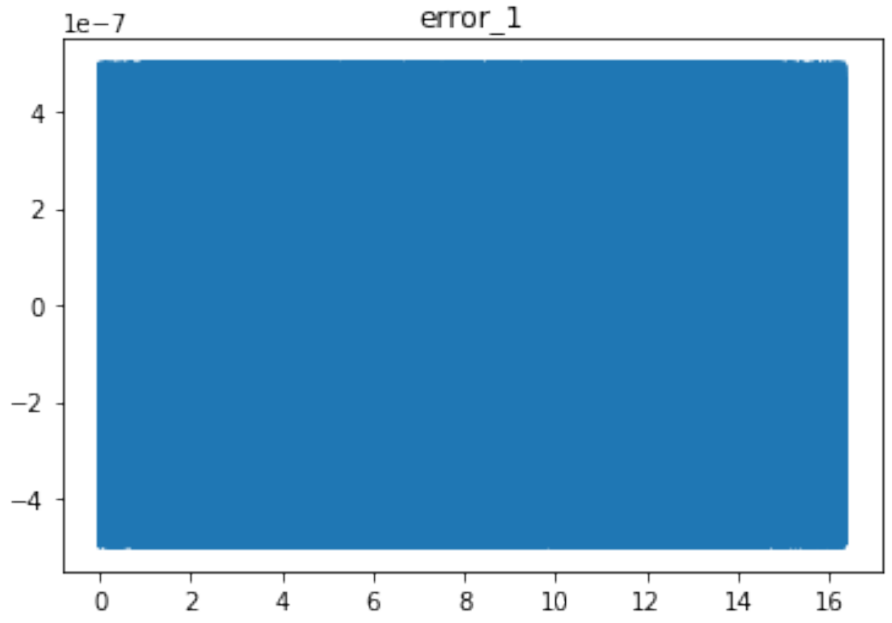


Chaos_y + Channel_2
40000
20000
0
−20000
−40000
0
2
4
6
8
10
12
14
16

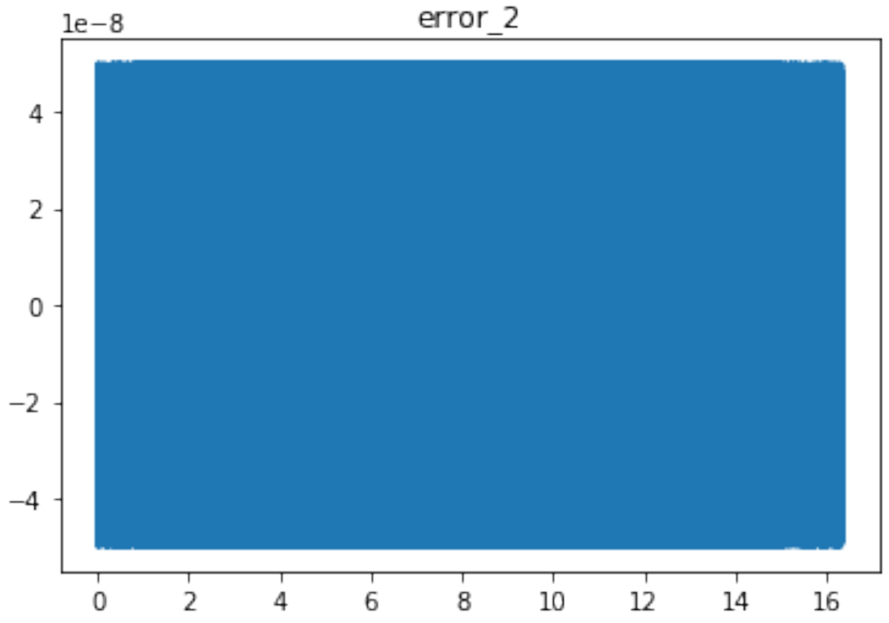


Chaos_y + Channel_2
40000
20000
0
−20000
−40000
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200

## 誤差分析



1e−7
error_1
4
2
0
−2
−4
0
2
4
6
8
10
12
14
16



1e−8
error_2
4
2
0
−2
−4
0
2
4
6
8
10
12
14
16

## 討論

- 使用者友善圖形界面設計

# 分工表

- 李岱庭：加密方、聲音播放
- 林希雲：加密方、投影片製作
- 饒孝節：解密方、圖形化對話框
- 許祐綸：解密方、程式碼簡化、整理
- 楊馥榕：混沌製造、資料收集