

实验 C9 混沌电路实验*

XXX[†]

(1. 中山大学物理学院, 广东广州 510275)

摘要:混沌学属于非线性动力学,混沌理论认为混沌是在确定的非线性动力学系统中,不需要附加任何随机因素,在一定条件下就会呈现出类似随机但又遵循一定规律的复杂动力学振荡行为。利用电子电路的方法可以实现混沌振荡现象,本系列实验利用 Python 数值模拟, Multisim 和 LTspice 软件仿真和 NI Virtualbench 交互式测量实际电路相结合的方法,观察若干种混沌现象,从理论和实践上还原了蔡氏和洛伦兹混沌电路。首先,本实验探究了不含电容蔡氏电路的不同混沌状态,从 Python 的数值模拟上看,第一种蔡氏电路可以得到直线、极限环、双吸引子、第一次和第二次单吸引子五种相图。通过 Multisim 仿真和全真电路均可以得到这五种相图。使用配对 T 检验对仿真和全真电路不同相图时同一电阻不同阻值进行分析,可以得到 t 值为-1.12901974098575, 对应 P 值为 0.322013731315691>0.05, 无统计学差异, 故得出结论仿真可以较好的模拟真实的第一种蔡氏混沌电路。其次, 我们对含电容的蔡氏混沌电路进行 Multisim 仿真, 分别出现直线、极限环、双吸引子和第一次单吸引子相图。通过搭建真实电路, 使用 NI Virtualbench , 同样可以出现这四种相图。经配对 T 检验得 t 值为-0.587714318298662, 对应 P 值为 0.598032391735953>0.05, 无统计学差异, 故得出结论仿真可以较好的模拟真实的第一种蔡氏混沌电路。最后, 我们使用 Python 对经典洛伦兹混沌方程进行模拟, 得到双吸引子的三维图、时域图和各平面投影图。使用 LTspice 仿真, 当 R2 取不同阻值时, 洛伦兹混沌电路的相图分别为直线、双吸引子和第一次单吸引子。若调节更多原件的参数, 可能可以得到更多种类的相图。

关键词:混沌、蔡氏混沌电路、洛伦兹混沌电路、Multisim、LTspice、NI VirtualBench

Experiment C9: Chaotic circuit experiment*

XXX¹

1. School of Physics, Sun Yat-sen University, Guangzhou 510275, China

Abstract: Chaos belongs to nonlinear dynamics. chaos theory believes that chaos is in the determination of nonlinear dynamics. In this system, there is no need to add any random factors, while under certain conditions will appear to be random. The chaotic oscillation phenomenon can be constructed using electronic circuit method. This series of experiments used Python numerical simulation, Multisim, LTspice software simulation and NI Virtualbench interactive measurement of real electricity. Chua's and Lorentz's chaotic circuits were restored theoretically and practically by observing some chaotic phenomena. First, this experiment explored the different chaotic states of the chua's circuit without capacitors. From the Python numerical simulation, the first Chua circuit can obtain straight lines, limit cycles, double attractors, first and second single attractors. These five phase diagrams can be obtained by Multisim simulation and full - truth circuit as well. The pair-t test was used to analyze different resistance values of the same resistance in different phase diagrams of simulation and full true circuits. It can be concluded that the T value is -1.12901974098575, and the corresponding P value is 0.322013731315691>0.05, which is of no statistical difference. Therefore, it can be concluded that the simulation can finely simulate the real first Chua's chaotic circuit. Secondly, we did Multisim simulation for Chua's chaotic circuit with capacitor, and the

*由中山大学物理学院陆佑堂提供器材和指导。

†作者简介: XXX, xxxxxxxx; E-mail: xxxx@mail2.sysu.edu.cn

phase diagrams of straight line, limit cycle, double attractor and first single attractor appeared respectively. Using NI Virtualbench, these four diagrams can also appear by building real circuits. The t-value obtained by paired T-test is -0.587714318298662 , and the corresponding P-value is $0.598032391735953 > 0.05$, showing no statistical difference. Therefore, it is concluded that the simulation can finely simulate the real second Chua's chaotic circuit. Finally, we used Python to simulate the classical Lorentz chaos equation. We got the 3d image, time domain image and plane projection image of the double attractor. When R2 takes different resistance values, The LTspice simulation phase diagrams of Lorentz chaotic circuit were linear, double attractor and first single attractor. More types of phase diagrams may be obtained by adjusting the parameters of more elements.

Key words: Chaos, Chua's chaotic circuit, Lorentz chaotic circuit, Multisim, LTspice, NI VirtualBench

1 引言

混沌学属于非线性动力学，起源于 20 世纪初。混沌理论认为混沌是在确定的非线性动力学系统中，不需要附加任何随机因素，在一定条件下就会呈现出类似随机但又遵循一定规律的复杂动力学振荡行为。进入 20 世纪 90 年代以来，混沌学与其它学科相互渗透，相互促进，在物理学、电子学、信息科学、生物学、生理学、振动学、控制学、还是天文学、气象学、经济学等学科都得到广泛研究和应用。著名科学家钱学森认为，混沌是宏观无序，微观有序的现象，混沌学是专门研究复杂非线性动力学现象运动规律的一门科学。

2 实验目的

利用电子电路的方法可以实现混沌振荡现象，本系列实验利用软件仿真和实际电路相结合的方法，观察若干种混沌现象，以学习混沌现象的基础知识，并掌握简单电路设计的方法。

对于蔡氏电路，首先用 Jupyter Notebook 进行编程，解混沌方程组并得到不同维度和平面投影的相图；其次使用 Multisim 软件仿真，得到不同相图的电阻数据；最后学习和使用 NI ELVIS 平台和 VirtualBench 的使用方法，完成蔡氏混沌电路的搭建和实验，掌握蔡氏混沌电路的工作原理。

掌握了蔡氏电路后，基于 Jupyter Notebook 平台，编程生成自选混沌电路吸引子图像；并且利用 LTspice [1] 仿真软件，仿真自选洛伦兹混沌电路。

3 介绍 [2]

3.1 混沌电路

一个确定的动力系统有三种常见的稳定状态，平衡态、周期振荡态和准周期振荡态。混沌振荡是一种不稳定的但有限的动力学振荡行为，局限在有限区域，轨道永不重复且具有遍历性。混沌振荡有如下几个特征：

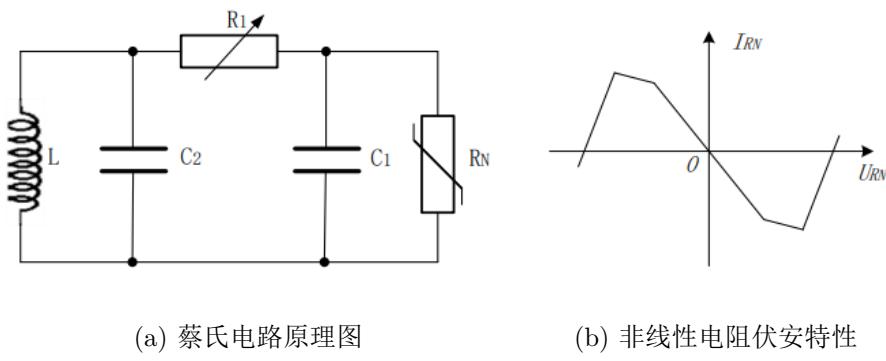
1. 混乱不是随机的，这意味着它由数学方程式控制。
2. 所有混沌系统对初始条件表现出极高的敏感性。因此，对混沌系统行为的长期预测是不可能的。（然而，良好的短程预测通常是可能的。）
3. 所有混沌系统都是非线性的（但并非所有非线性系统都是混沌系统）。
4. 所有由一组自主微分方程描述的混沌系统至少是三维的。（维度数是方程式中描述系统所需的变量数。）
5. 在相空间中为混沌系统追踪的轨迹永远不会交叉或接触。如果是这样，变量的值将重复自身，并且未来的值将是可预测的。

6. 两个相邻轨迹的发散速率取决于混沌系统的“Lyapunov 指数”。Lyapunov 指数的个数等于相空间的维数（自变量）。

Lyapunov 指数可以是正数或负数，但如果系统混乱，则其中至少有一个指数必须是正数。正指数越大，对初始条件系统越敏感，轨迹发散得越快。这样可以缩短系统行为的可预测性时间。

3.2 蔡氏电路

20世纪60年代Lorenz在实验中发现第一个混沌吸引子的Lorenz系统，1984年华裔学者蔡少棠提出了著名的“蔡氏电路”，通过计算机和电子电路实验研究，证实了“蔡氏电路”是一种自激振荡电路，在一定参数条件下，能产生各种分岔，单旋涡和双旋涡吸引因子等丰富和复杂的非线性现象和混沌动力学行为。之后相继发现了许多新型混沌系统，如分数阶系统，多翼混沌系统，超混沌系统及恒Lyapunov指数系统等。在混沌现象研究中，蔡氏电路是被广泛用作混沌实验教学的经典电路。原理图如图3.1a所示，由一个线性电感L，两个线性电容C₁、C₂，一个线性电阻R₁和一个非线性电阻R_N构成。蔡氏电路是一个结构简单的三阶自治动态系统，能产生丰富的混沌现象。其中电感L和电容C₂构成一个LC振荡电路，非线性电阻R_N是有源非线性的分段线性电阻，与电容C₁并联滤波电路将振荡器产生的正弦信号移相输出，线性电阻R₁调节C₁和C₂相位差，并消耗能量。非线性电阻R_N伏安特性如图3.1b所示。



(a) 蔡氏电路原理图

(b) 非线性电阻伏安特性

图 3.1

得到电路的非线性动力学方程如下所示：

$$\begin{cases} C_1 \cdot \frac{dU_{C1}}{dt} = \frac{1}{R_1} \cdot (U_{C2} - U_{C1}) - f(U_{RN}) \\ C_2 \cdot \frac{dU_{C2}}{dt} = i_L - \frac{1}{R_1} \cdot (U_{C2} - U_{C1}) \\ L \cdot \frac{di_L}{dt} = -U_{C2} \end{cases}$$

其中, U_{C1} 是电容 C₁ 上的电压, U_{C2} 是电容 C₂ 上的电压, i_L 是电感上电流, 由于 R_N 是非线性电阻, $f(u_{RN})$ 是非线性函数, 上述方程组没有解析解, 且方程组右端不显含时间变量的微分方程组, 构成三阶自治动态系统。

归一化后的蔡氏混沌电路方程为:

$$\begin{cases} \frac{dx}{d\tau} = \alpha \cdot (y - x) - f(x) \\ \frac{dy}{d\tau} = x - y + z \\ \frac{dz}{d\tau} = -\beta y \end{cases}$$

$$f(x) = m_b \cdot x + \frac{1}{2}(m_a + m_b) \cdot (|x + 1| - |x - 1|)$$

3.3 洛伦兹混沌电路

洛伦兹混沌电路的模拟计算机电路是由哈佛大学的 Paul Horowitz 开发的。该解决方案执行一个轨迹，绘制在三维空间中，既不可预测也不可随机地盘旋，占据一个称为吸引子的区域。在一个模拟电子电路中实现这些方程是相当容易的，只需要 3 个运放（每个运放同时进行积分和求和）和两个模拟乘法器（形成 xy 和 xz 的乘积）。运算放大器被连接成积分器，在输入处对构成每个导数的各项求和。电阻器的值被缩放到 1 兆欧，因此， R_3 对变量 x 的权重为 28 ($1M/35.7k$)；它与 $-y$ 和 $-xz$ 结合，每个都有单位重量。其电路如图3.2所示：

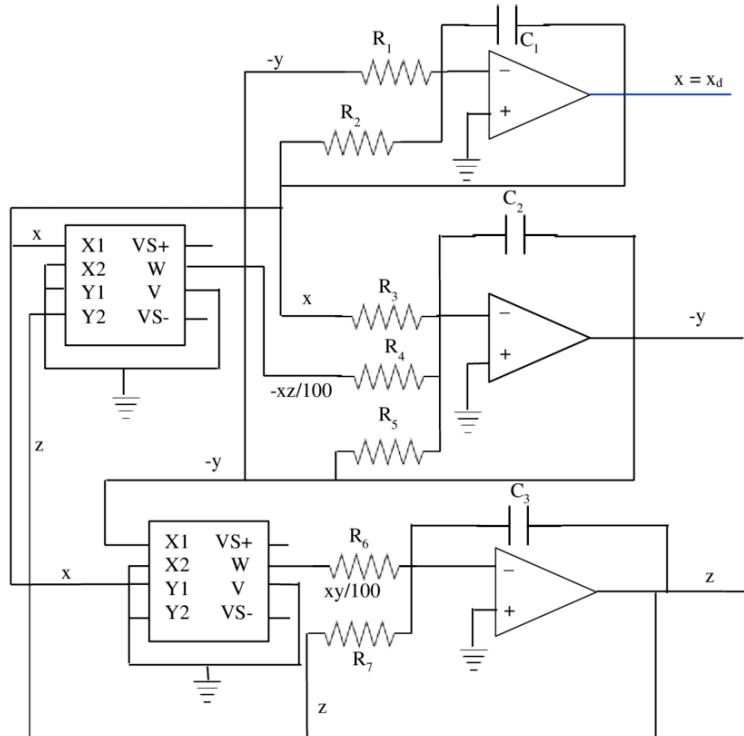


图 3.2: 洛伦兹混沌电路

3.4 Multisim

1988 年，加拿大 IIT(Interactive Image Technologies) 公司推出 EWB (Electronics Work Bench) 套件，它是用于电子线路设计和仿真的 EDA 软件。该套件可对模拟、数字、模拟/数字混合电路仿真，使用界面直观、方便，同时具有强大分析功能。随着科技进步，IIT 公司随后就推出了更高版本的 EWB 软件。在推出 EWB6.0 时，将套件名称改为 Multisim，意为多功能仿真软件，即 Multisim2001。2005 年，美国 NI(National Instrument) 将 IIT 公司收购，软件改名为 NI Multisim，随后相继推出多个版本，最新的版本为 NI Multisim2016。由于版权的原因，本实验将采用 NI Multisim12.0 版软件。

Multisim12 仿真软件是电子线路仿真软件，基于工业标准 SPICE 仿真，以获得最优化的设计。推出后，被广泛用于电路设计和电子教学中，Multisim12 可以对电路原理、电路功能进行仿真测试和分析功能。有大量与实际元件对应的元件模型，提供了一个完整的集成化设计环境，使仿真设计结果更精确、可靠、具有实用性，并在功能和操作方法上有较大的改进，电路分析手段完备，可实现大部分硬件电路实验的功能。

3.5 NI VirtualBench

NI VirtualBench 是用 LabVIEW 编程控制的多功能仪器，集成了混合信号示波器、函数发生器 (FGEN)、数字万用表、数字 I/O 口和直流电源等五种仪器的功能。为了方便使用，设备厂家已编制了 VirtualBench 虚拟

仪器测控软件，可以直接使用。若需要用 LabVIEW 对该平台进行编程控制，进行二次开发，则需要 LabVIEW 2015 及以上的版本，测控计算机还需要同时安装 VirtualBench18.0 驱动程序。

3.6 LTspice

LTspice 是一款高性能 Spice III 仿真器、原理图捕获和波形查看器，具有增强功能和模型，可简化开关稳压器的仿真。与普通的 Spice 模拟器相比，Spice 的增强功能使开关稳压器的仿真速度极快，使用户能够在短短几分钟内查看大多数开关稳压器的波形。

下载地址和实验方法详细可参考:[LTspice](#)

4 实验仪器

ELVIS II+ 平台及原型板一套，NI VirtualBench-8012 一体化仪器，实验测控用计算机（安装了 Multisim、LabVIEW、ELVIS Launcher、LTspice 和 NI VirtualBench 软件），电子元器件若干和面包板两块，螺丝批和绝缘镊子各 1 把，导线若干。

5 实验结果

5.1 蔡氏电路（无电容）

5.1.1 数值模拟

在 vscode 编译器中使用 Jupyter Notebook, 用 Python 语言编写蔡氏电路的微分方程：

$$\begin{cases} \frac{dx}{d\tau} = \alpha \cdot (y - x) - f(x) \\ \frac{dy}{d\tau} = x - y + z \\ \frac{dz}{d\tau} = -\beta y \end{cases}$$

$$f(x) = m_b \cdot x + \frac{1}{2}(m_a + m_b) \cdot (|x + 1| - |x - 1|)$$

通过调节参数 M_a 、 M_b 、 α 和 β 产生不同图像的相图，并且作出三维相图，时域图和平面投影图。

- (1) $M_a=-2.0$, $M_b=-1.3$, $\alpha=26$, $\beta=12$, 得到的相图为直线, 如图[5.1](#)所示。
- (2) $M_a=-0.5$, $M_b=1$, $\alpha=13$, $\beta=13$, 得到的相图为极限环, 如图[5.2](#)所示。
- (3) $M_a=-1.1$, $M_b=0.7$, $\alpha=21$, $\beta=36$, 得到的相图为双吸引子, 如图[5.3](#)所示。
- (4) $M_a=-1.1$, $M_b=-0.7$, $\alpha=21$, $\beta=31.3$, 得到的相图为第一次单吸引子, 如图[5.4](#)所示。
- (5) $M_a=-1.1$, $M_b=-0.7$, $\alpha=21$, $\beta=150$, 得到的相图为第二次单吸引子, 如图[5.5](#)所示。

5.1.2 Multisim 仿真

使用 Multisim 软件搭建仿真蔡氏电路，调节 R7 阻值，测量 V1、V2 处电压得到不同相图。

- (1) $R7 = 0\Omega$

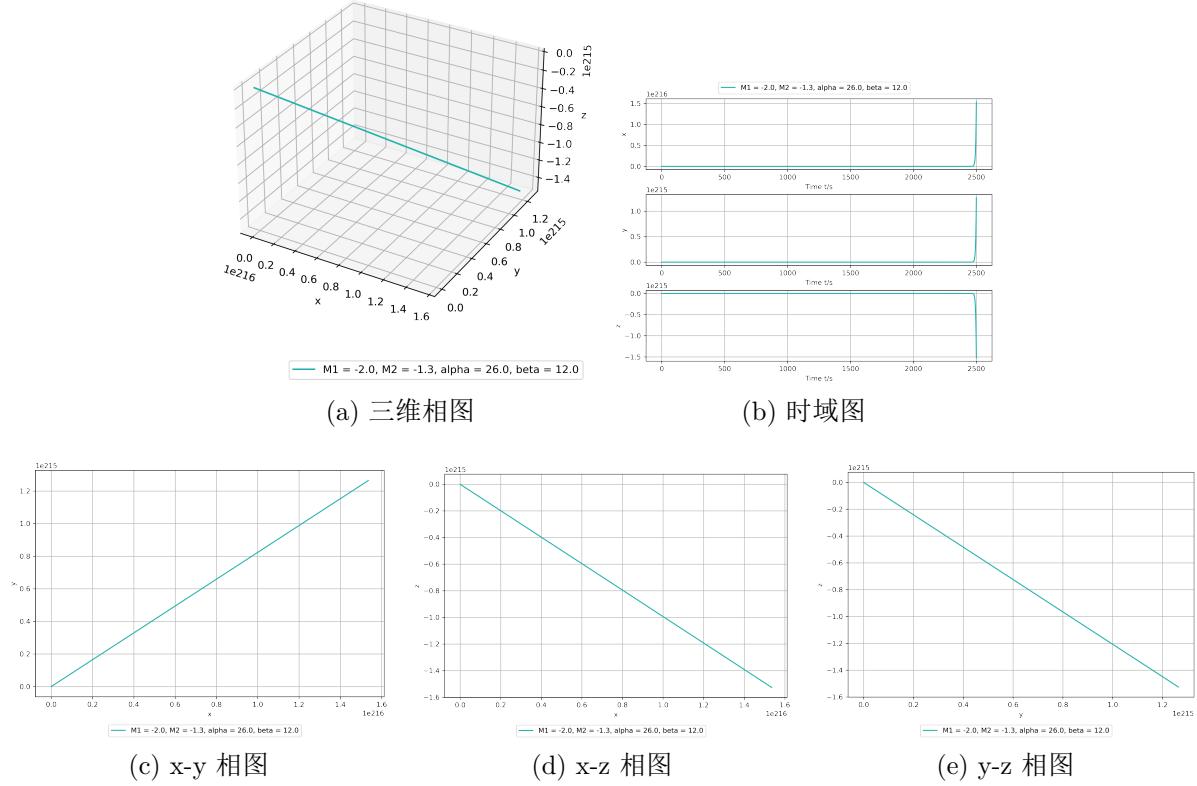


图 5.1: 直线

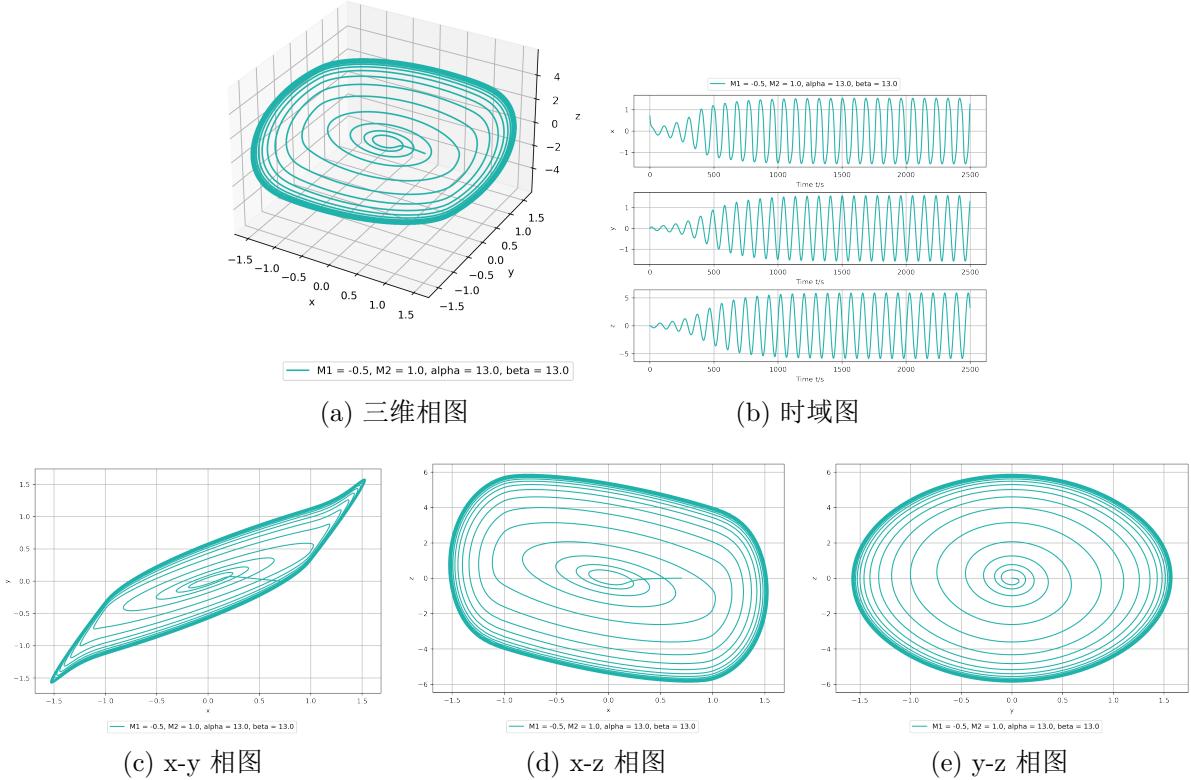


图 5.2: 极限环

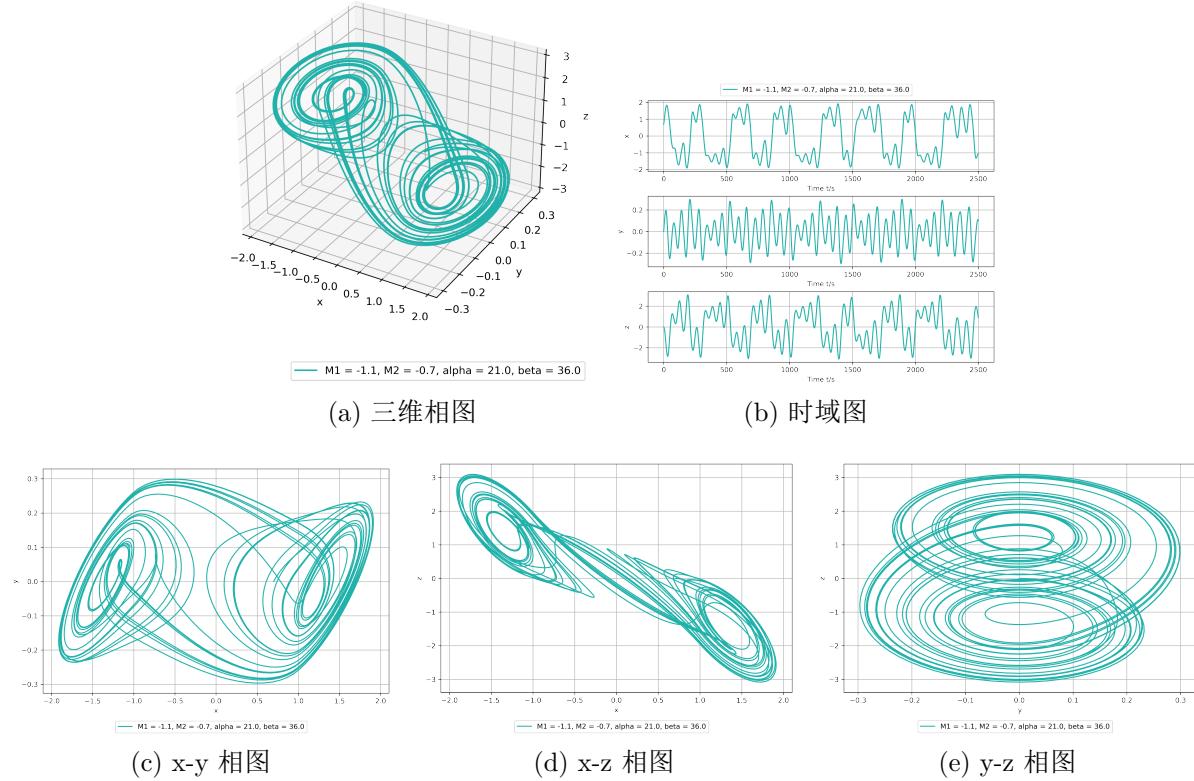


图 5.3: 双吸引子

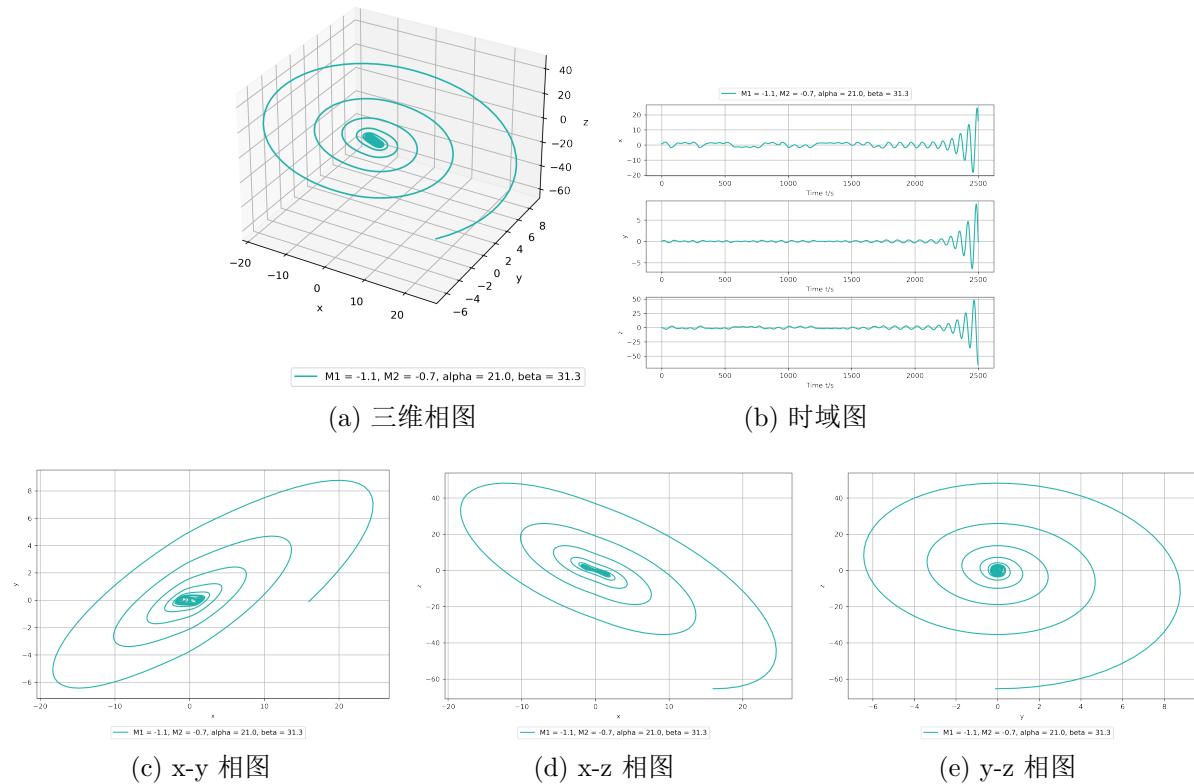


图 5.4: 第一次单吸引子

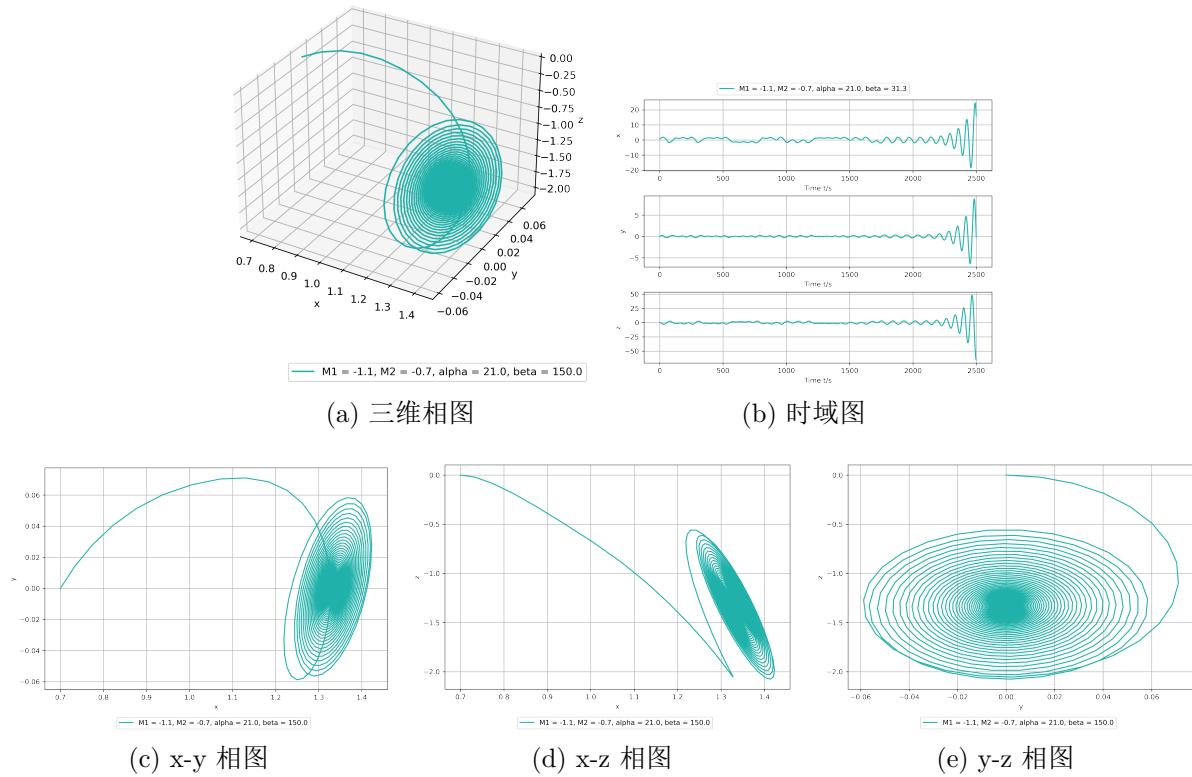


图 5.5: 第二次单吸引子

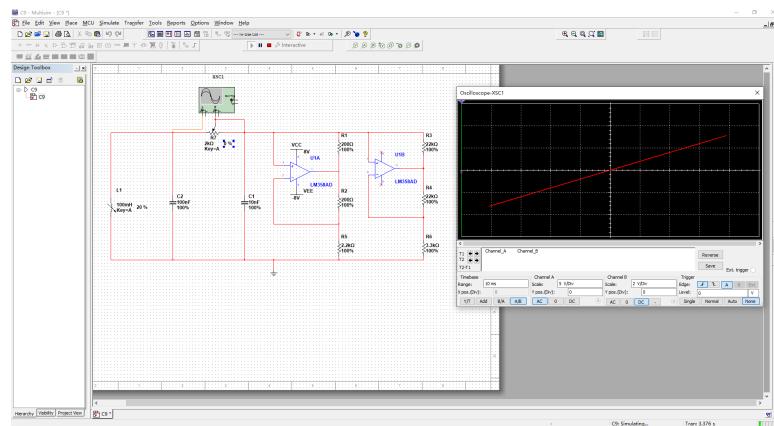


图 5.6: 直线

(2) $R7 = 1536\Omega$

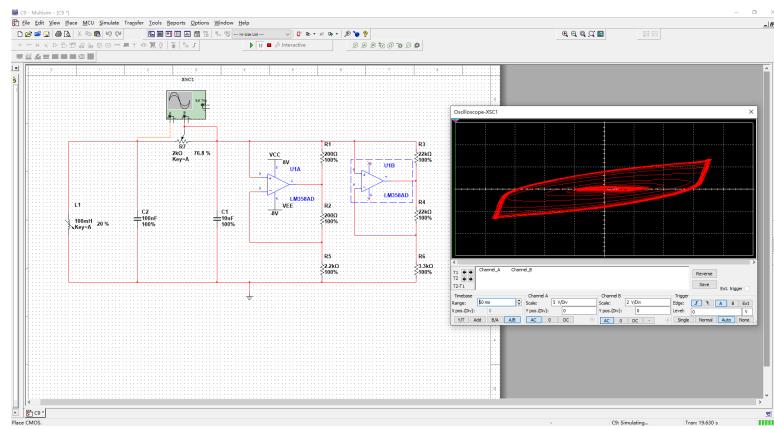


图 5.7: 极限环

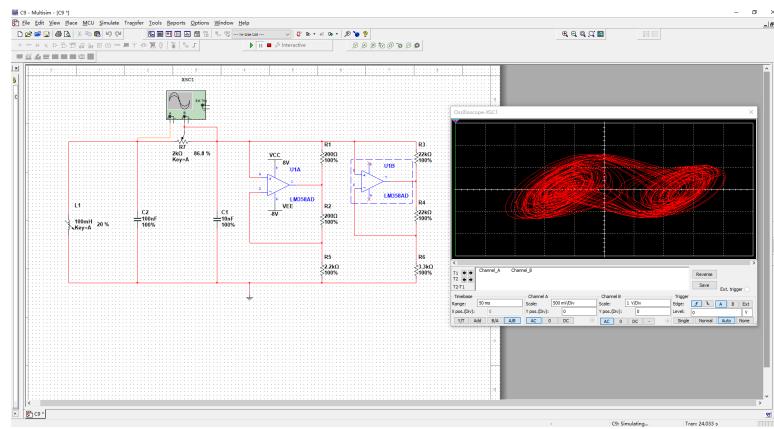
(3) $R7 = 1736\Omega$ 

图 5.8: 双涡旋吸引因子

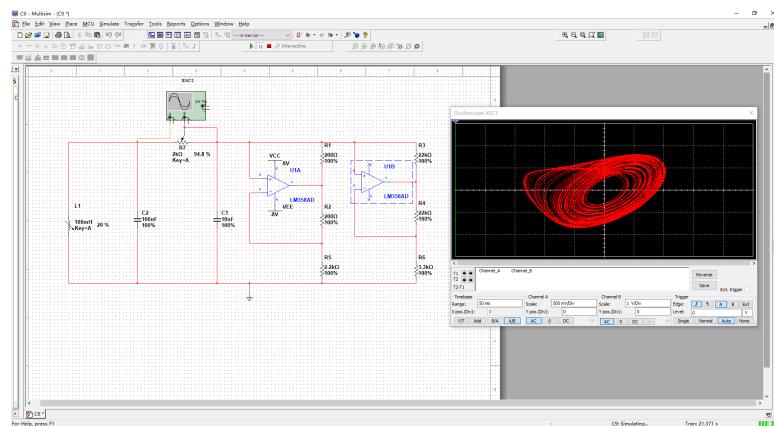
(4) $R7 = 1896\Omega$ 

图 5.9: 第一次单涡旋吸引因子

(5) $R7 = 1920\Omega$

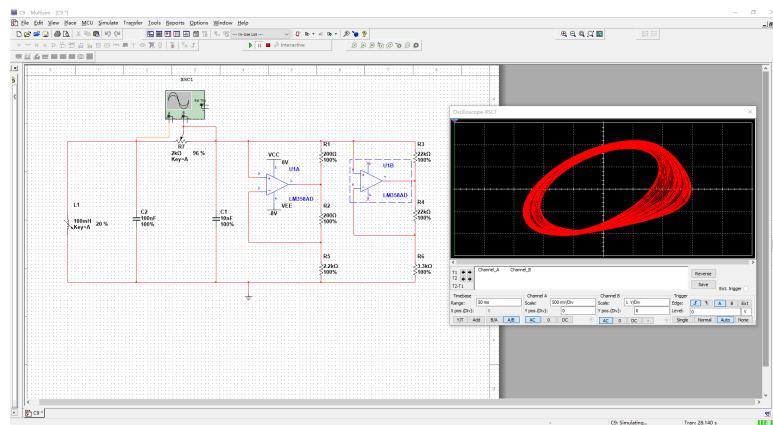
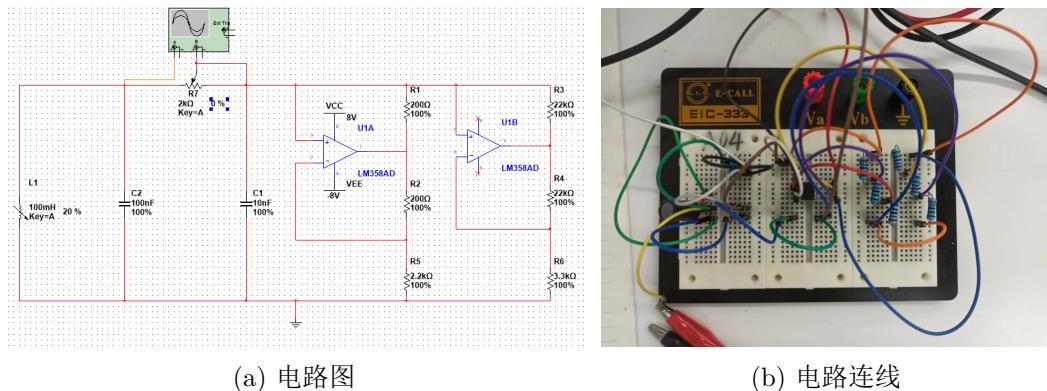


图 5.10: 第二次单涡旋吸引因子

5.1.3 全真电路

按图5.11a在面包板上搭建蔡氏电路，全真电路如图5.11b所示。



(a) 电路图

(b) 电路连线

图 5.11: 实验图

改变滑动变阻器阻值，测量 V1 和 V2 处电压，可以得到不同相图。

(1) $R7 = 0\Omega$

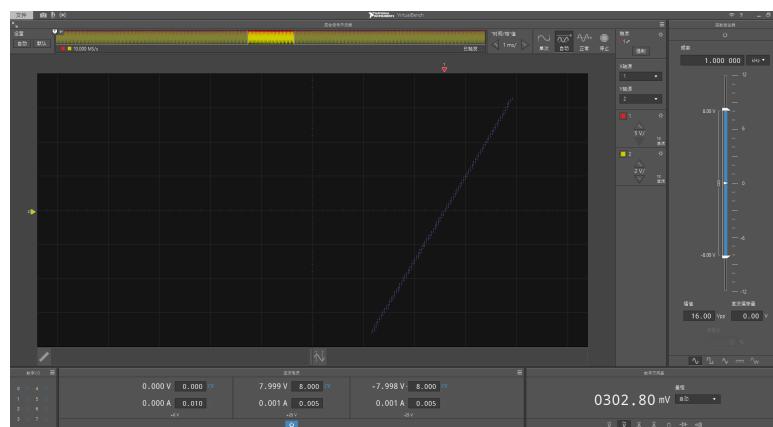


图 5.12: 直线

(2) $R7 = 1498.20\Omega$

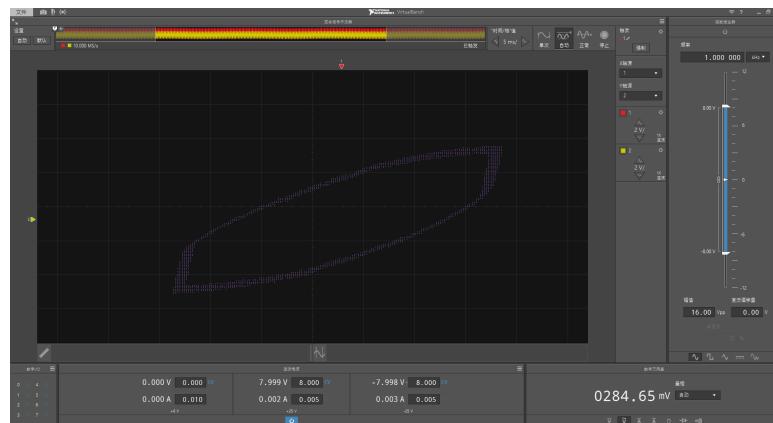


图 5.13: 极限环

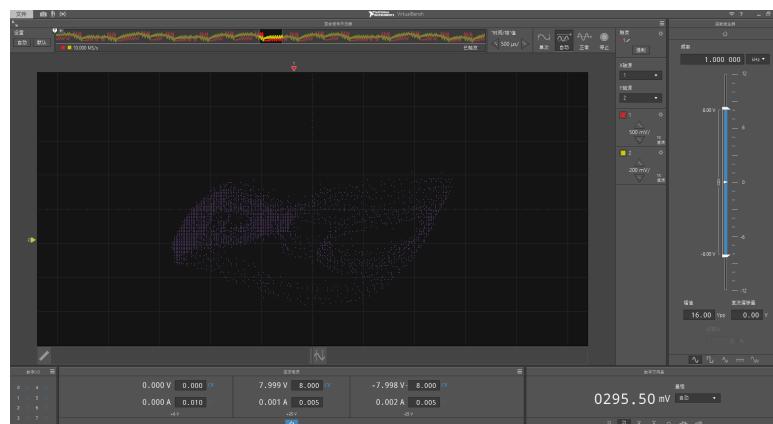
(3) $R7 = 1880.32\Omega$ 

图 5.14: 双涡旋吸引因子

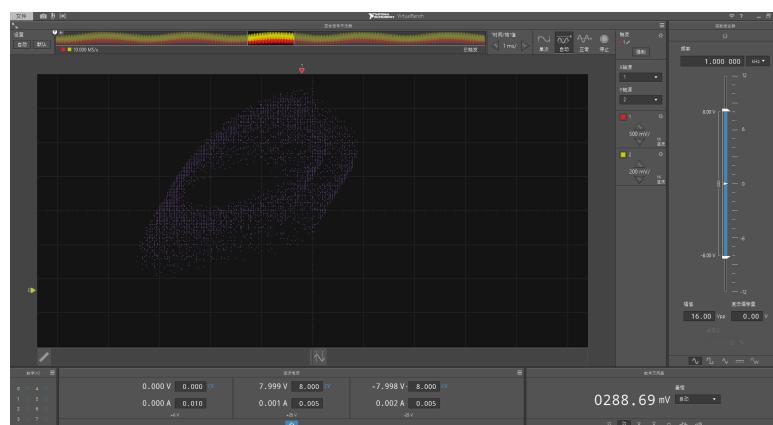
(4) $R7 = 1907.20\Omega$ 

图 5.15: 第一次单涡旋吸引因子

(5) $R7 = 1979.10\Omega$

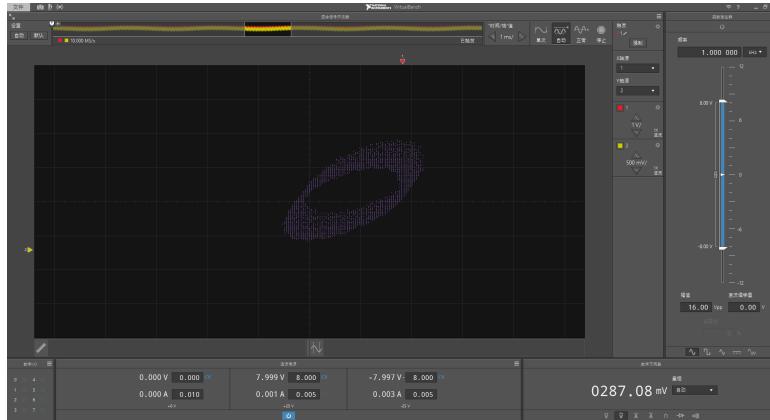


图 5.16: 第二次单涡旋吸引因子

数据分析

仿真和实际电路不同相图时 R7 阻值见表 1:

实验方式	直线	极限环	双吸引子	第一次单吸引子	第二次单吸引子
仿真电路	0	1536	1736	1896	1920
全真电路	0	1498.20	1880.32	1907.20	1979.10

表 1: 仿真和实际电路不同相图时 R7 阻值 (单位: Ω)

经配对 T 检验得 t 值为 -1.12901974098575 , 对应 P 值为 $0.322013731315691 > 0.05$, 无统计学差异, 故得出结论仿真可以较好的模拟真实的第一种蔡氏混沌电路。

5.2 蔡氏电路 (带电容)

5.2.1 Multisim 仿真

使用 Multisim 软件搭建如下图左侧所示的电路, 改变 R_k 阻值可以得到不同相图。

(1) $R_k = 0\Omega$

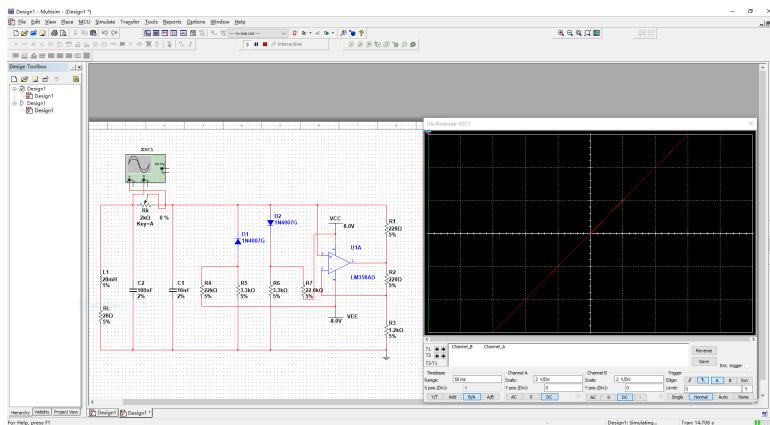


图 5.17: 直线

(2) $R_k = 1040\Omega$

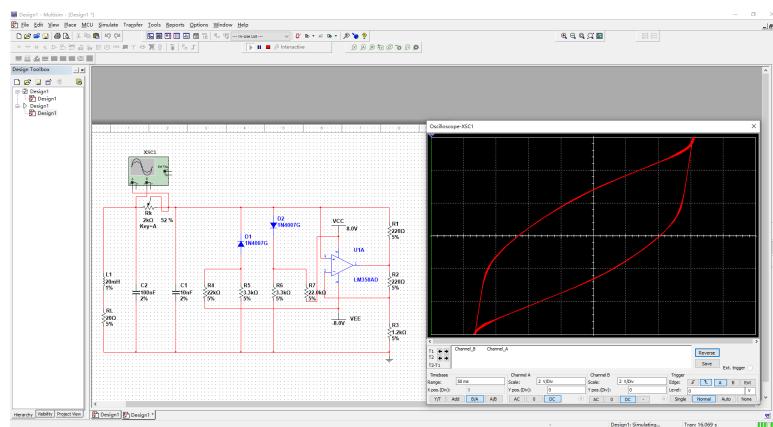


图 5.18: 极限环

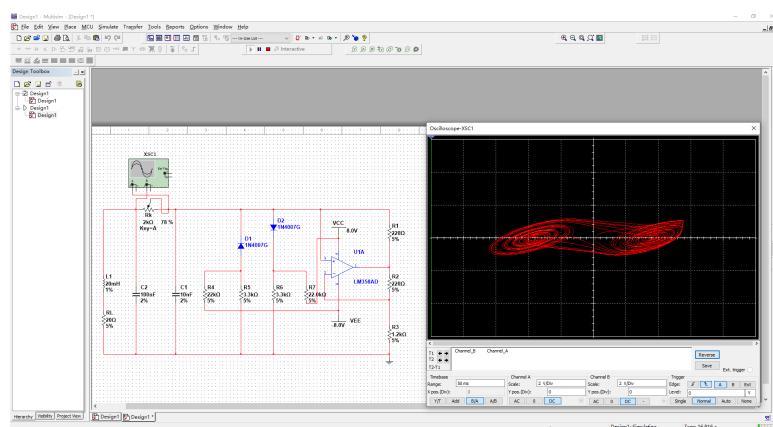
(3) $Rk = 1560\Omega$ 

图 5.19: 双涡旋吸引因子

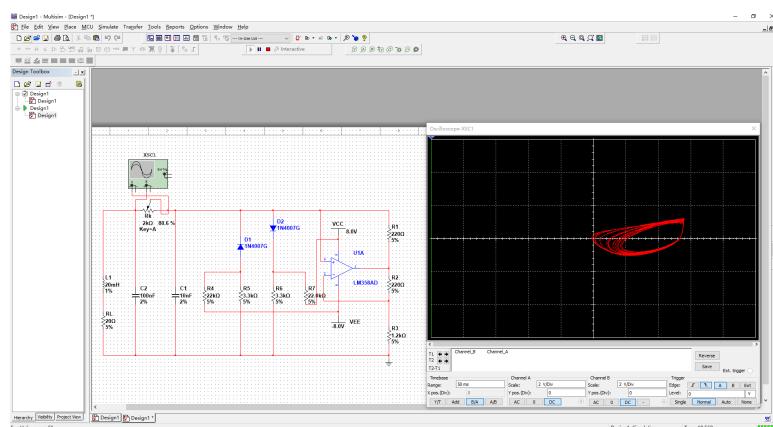
(4) $Rk = 1601.2\Omega$ 

图 5.20: 第一次单涡旋吸引因子

调节 Rk 过程中未出现第二次单涡旋吸引因子。

5.2.2 全真电路

按图5.21a在面包板上搭建另一种蔡氏电路，全真电路如图5.21b所示。

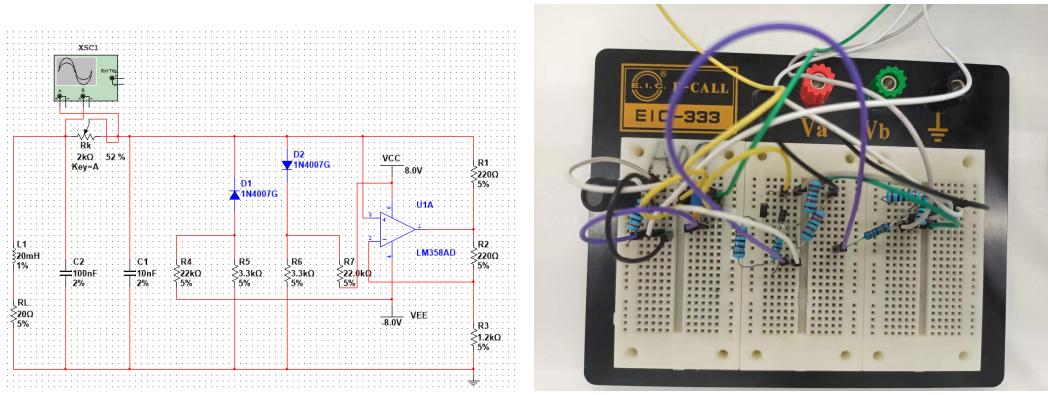


图 5.21: 实验图

改变滑动变阻器阻值，测量 V1 和 V2 处电压，仍可以得到不同相图。

(1) $R7 = 0\Omega$

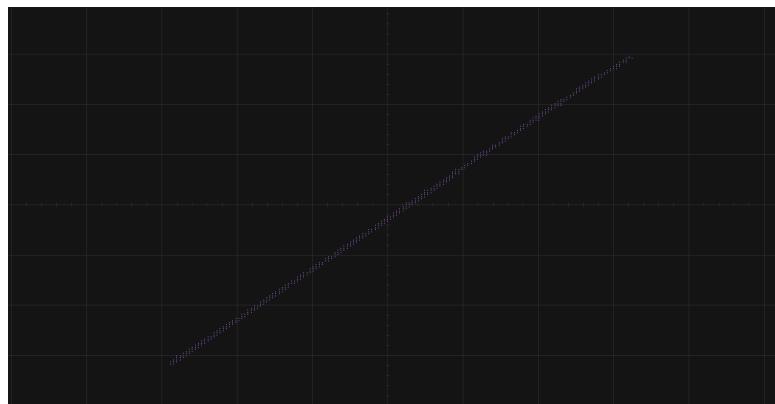


图 5.22: 直线

(2) $R7 = 1048\Omega$

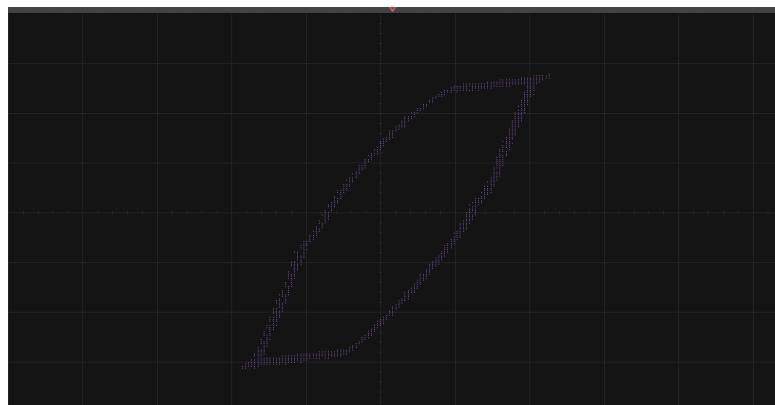


图 5.23: 极限环

(3) $R7 = 1563.2\Omega$

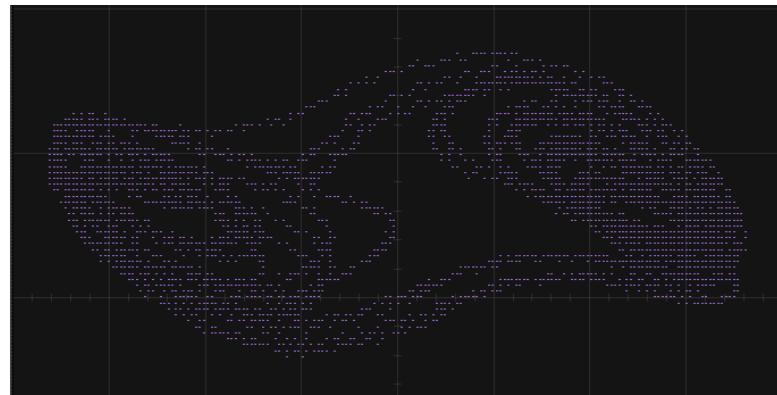


图 5.24: 双涡旋吸引因子

(4) $R7 = 1661\Omega$

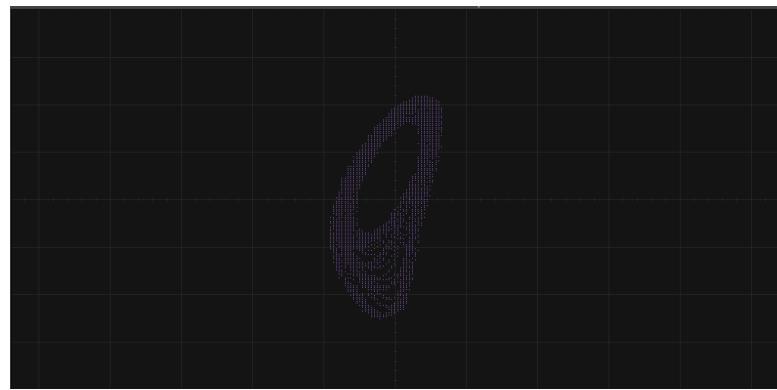


图 5.25: 第一次单涡旋吸引因子

与仿真结果类似，调节 R_k 过程中也未出现第二次单涡旋吸引因子。

数据分析

仿真和实际电路不同相图时 R_k 阻值见表 2:

实验方式	直线	极限环	双吸引子	第一次单吸引子
仿真电路	0	1040	1560	1601.2
全真电路	0	1018	1563.2	1661

表 2: 仿真和实际电路不同相图时 R_k 阻值 (单位: Ω)

经配对 T 检验得 t 值为 -0.587714318298662 , 对应 P 值为 $0.598032391735953 > 0.05$, 无统计学差异, 故得出结论仿真可以较好的模拟真实的第二种蔡氏混沌电路。

5.3 洛伦兹混沌电路

5.3.1 数值模拟

在 vscode 编译器中使用 Jupyter Notebook, 用 Python 语言编写洛伦兹混沌方程:

$$\begin{cases} \frac{dx}{d\tau} = -10x + 10y \\ \frac{dy}{d\tau} = 28x - y - xz \\ \frac{dz}{d\tau} = -\frac{8}{3}z + xy \end{cases}$$

此时系统处在混沌状态, 该三维图5.26a在 XY 平面的投影如图5.26c所示, 在 XZ 平面的投影如图5.26d所示, 在 YZ 平面的投影如图5.26e所示, 时域图为图5.26b, 可以观察到经典的洛伦兹双吸引子(蝴蝶)曲线。

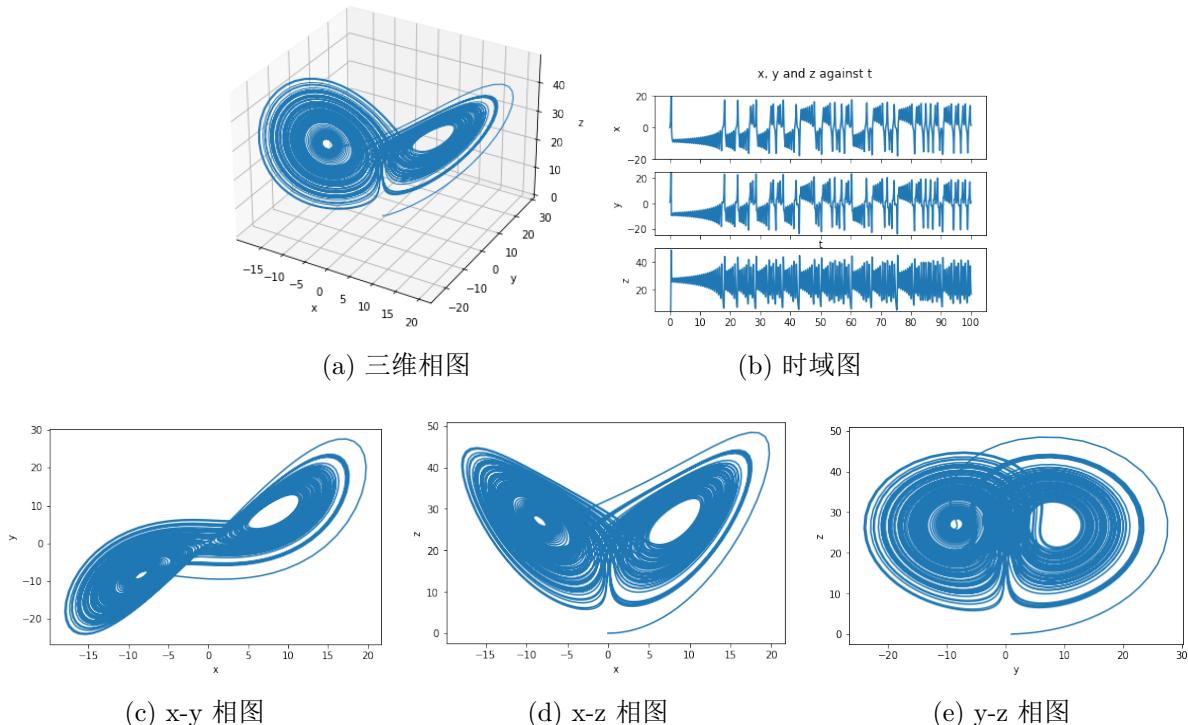


图 5.26: 数值模拟相图

5.3.2 洛伦兹混沌电路 LTspice 仿真

使用 LTspice 软件搭建洛伦兹混沌电路, 调节 R2 阻值可以得到不同相图。

(1) 直线

此时 $R2 = 1\Omega$, 仿真电路如下图:

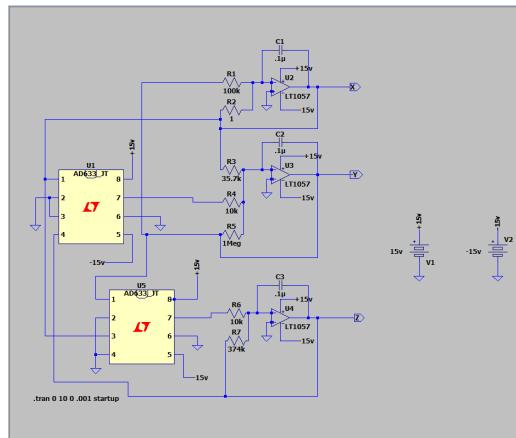


图 5.27: 直线相图仿真电路

在 XY 平面的投影如图 5.28a 所示，在 XZ 平面的投影如图 5.28b 所示，在 YZ 平面的投影如图 5.28c 所示，时域图为图 5.28d。

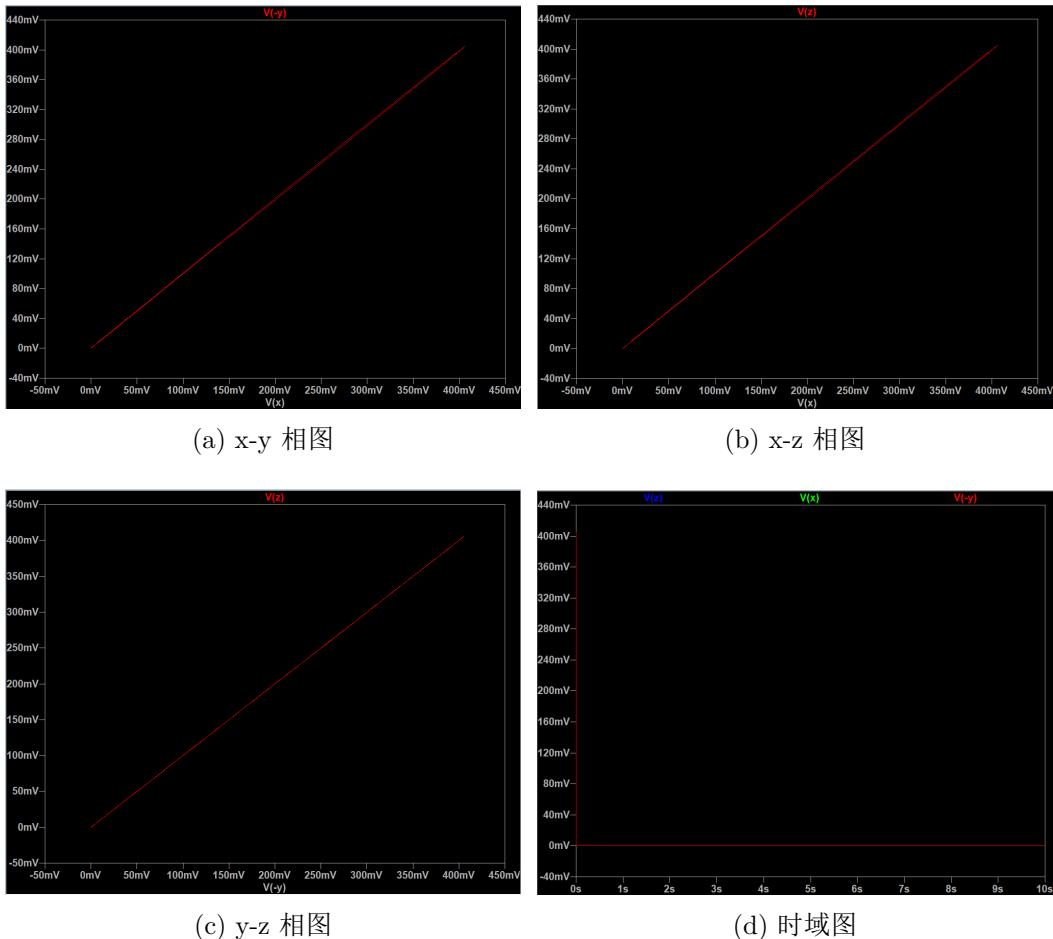


图 5.28: 直线仿真

(2) 双涡旋吸引因子

此时 $R2 = 100k\Omega$, 仿真电路如下图:

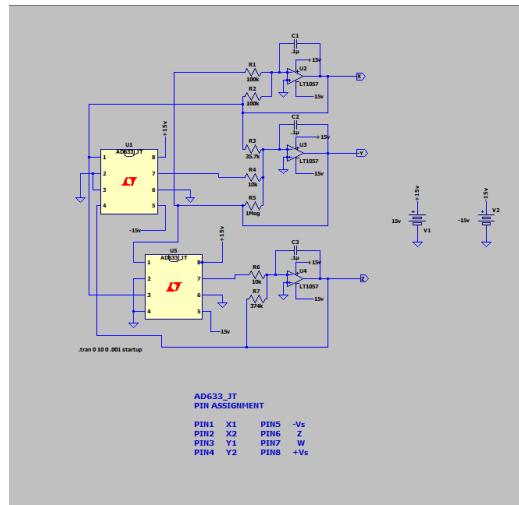


图 5.29: 双吸引子仿真电路

在 XY 平面的投影如图 5.30a 所示，在 XZ 平面的投影如图 5.30b 所示，在 YZ 平面的投影如图 5.30c 所示，时域图为图 5.30d。

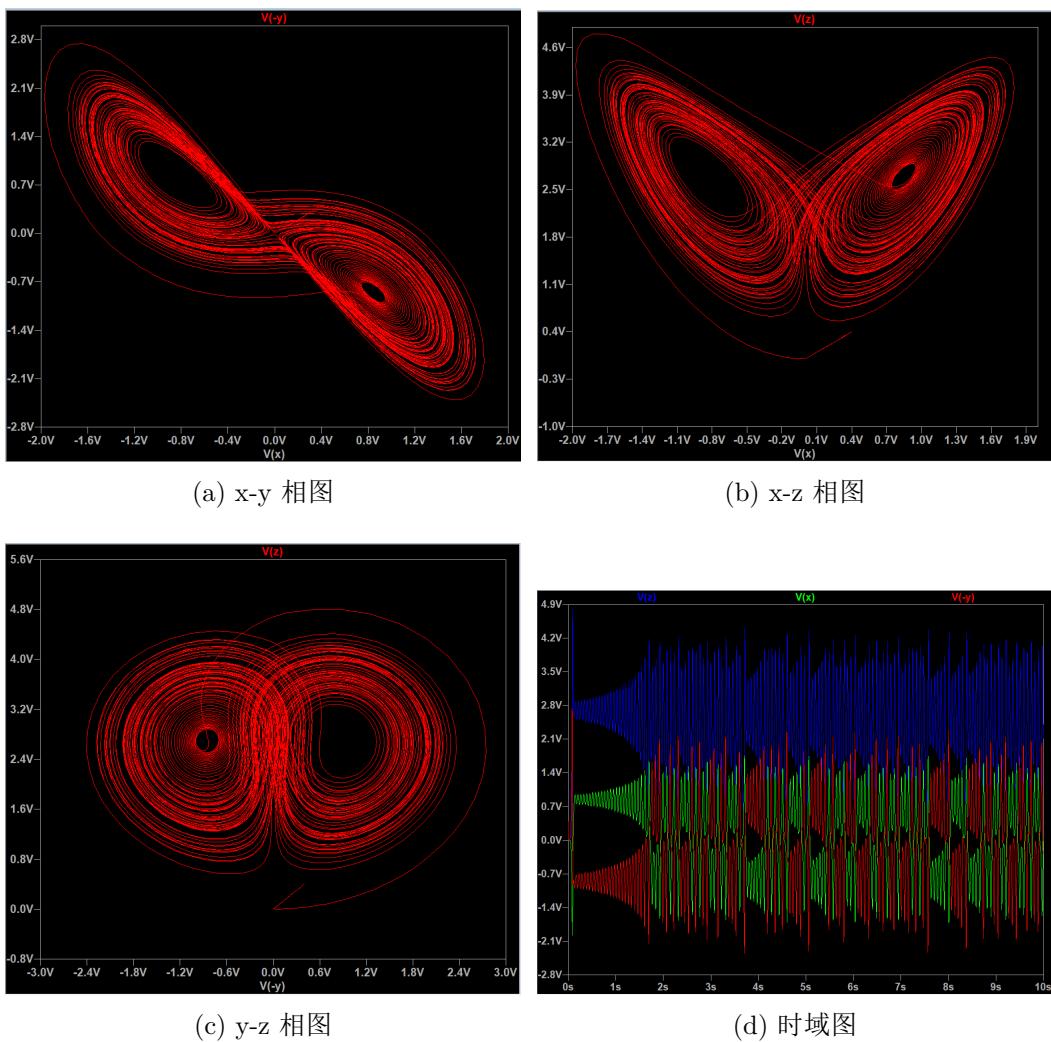


图 5.30: 双吸引子仿真

(3) 单涡旋吸引因子

此时 $R_2 = 640k\Omega$, 仿真电路如下图:

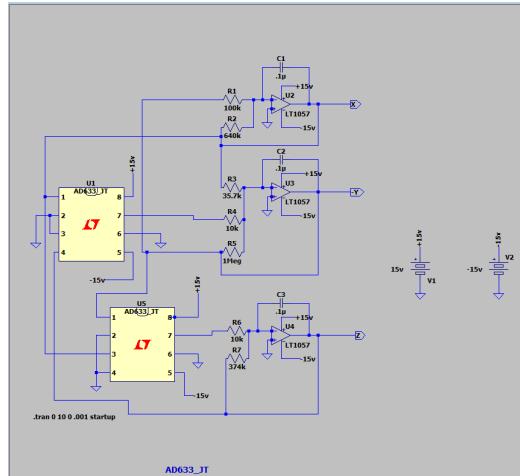
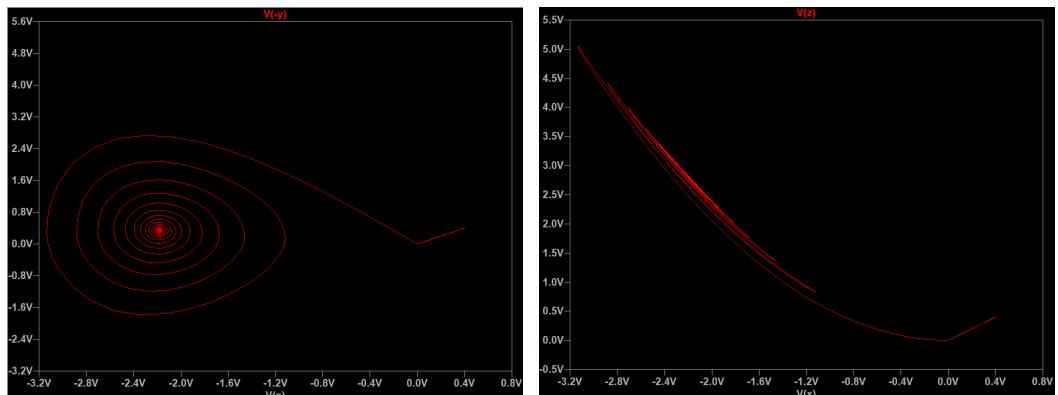


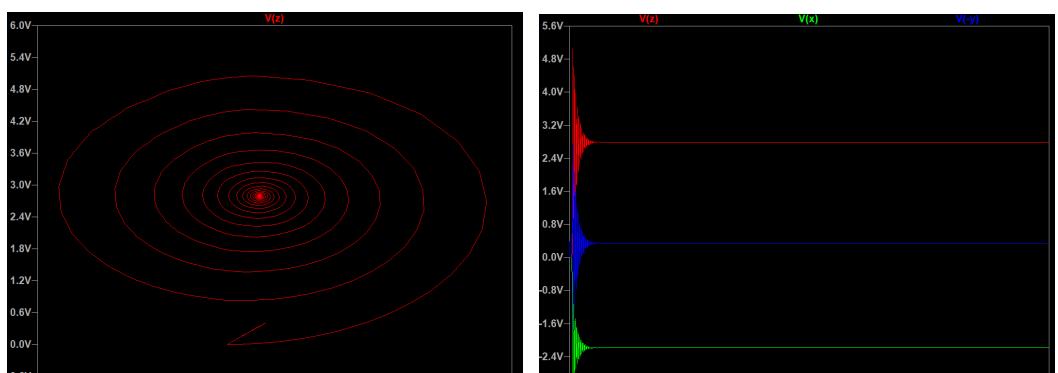
图 5.31: 单吸引子仿真电路

在 XY 平面的投影如图 5.32a 所示, 在 XZ 平面的投影如图 5.32b 所示, 在 YZ 平面的投影如图 5.32c 所示, 时域图为图 5.32d。



(a) x-y 相图

(b) x-z 相图



(c) y-z 相图

(d) 时域图

图 5.32: 单吸引子仿真

6 讨 论

在蔡氏混沌电路的仿真电路和全真电路中，并没有测量非线性伏安特性曲线电阻的阻值变化，故数值仿真时缺少关键数据 M_a 和 M_b 。此处数值计算并不能很好的与仿真电路和全真电路对应，故调节参数 M_a 、 M_b 、 α 和 β ，得到不同相图，仅表示蔡氏电路混沌状态时相图有直线、极限环、双吸引子、第一次单吸引子和第二次单吸引子五种，并不参与与仿真电路和全真电路的对比之中。若实验中测量非线性电阻曲线，并拟合成函数，则可带入完成精确的数值模拟。

在第二种蔡氏电路的模拟中，其混沌方程与第一种共用，但是由于非线性电阻的改变，第二种蔡氏电路混沌相图并不能出现第二次单吸引子。

在洛伦兹混沌电路的模拟中，数值模拟仅代入了最常见的双吸引子图像参数，仿真时也仅仅调节了 R2 的阻值，出现了直线，双吸引子和第一次单吸引子的图像，若调节不同电阻或电容，有可能出现更多不同种类的相图。

关于误差分析，由于数值模拟和仿真建立在精确的计算机计算上，所以最主要的误差来源于全真电路。首先选用的电阻和电容阻值与计算仿真略有误差，其次电路的接线方法会在一定程度上影响电路的灵敏性，例如电路中布置大量导线会增大电路断路的几率。

7 结 论

本实验采用数值计算、仿真实验和全真电路三种方式从理论和实践上还原了混沌电路。

首先，本实验探究了不含电容蔡氏电路的不同混沌状态，从 Python 的数值模拟上看，第一种蔡氏电路可以得到直线、极限环、双吸引子、第一次和第二次单吸引子五种相图。通过 Multisim 仿真，分别取 R7 为 0、1536、1736、1896 和 1920Ω ，可以得到这五种相图。通过搭建真实电路，使用 NI Virtualbench 可以发现当滑动变阻器阻值为 0、1498.2、1880.32、1907.20、 1979.10Ω 时分别出现五种相图。使用配对 T 检验可以得到 t 值为 -1.12901974098575，对应 P 值为 $0.322013731315691 > 0.05$ ，无统计学差异，故得出结论仿真可以较好的模拟真实的第一种蔡氏混沌电路。

其次，我们对含电容的蔡氏混沌电路进行 Multisim 仿真，发现 R_k 取 0、1040、1560、 1601.2Ω 时分别出现直线、极限环、双吸引子和第一次单吸引子相图。通过搭建真实电路，使用 NI Virtualbench，可以发现当滑动变阻器阻值为 0、1018、1563.2 和 1661Ω 时出现这四种相图。经配对 T 检验得 t 值为 -0.587714318298662，对应 P 值为 $0.598032391735953 > 0.05$ ，无统计学差异，故得出结论仿真可以较好的模拟真实的第二种蔡氏混沌电路。

最后，我们使用 Python 对经典洛伦兹混沌方程进行模拟，得到双吸引子的三维图、时域图和各平面投影图。使用 LTspice 仿真，当 R_2 为 1、100k、 $640k\Omega$ 时，洛伦兹混沌电路的相图分别为直线、双吸引子和第一次单吸引子。若调节更多原件的参数，可能可以得到更多种类的相图。

参考文献

- [1] DEVICES A. Handbook of LTspice[Z]. <https://www.analog.com/cn/design-center/design-tools-and-calculators/ltpice-simulator.html>. 2022 (引用页: 2).
- [2] SHEN H. General Physics Laboratory[M]. Science press, 2015 (引用页: 2).
- [3] 李征. 混沌电路的控制及其在通信中的应用[J]. 西安理工大学, 2007, 01(TN918) (引用页: 22).
- [4] 童勤业. 混沌电路的温度特性及其在温度测量中的应用[J]. 电路与系统学报, 2003, (04):21-24 (引用页: 22).

附录 A

【思考题】

1. 检索资料，简述混沌电路的可能应用。要求给出参考文献。

保密通信

由于混沌振荡频谱既有貌似随机又有宽频带连续频谱，在无线电通信上很适合用于保密通信与扩频通信。要利用混沌进行保密通信，要求保密通信双方必须有完全相同的非线性电路（混沌电路），这样才能达到混沌同步，从而实现保密信号从发射机的编码到接收机解码的全过程。否则无法达到混沌同步，无法解密信息与信号，也就无法进行保密通信。混沌通信方式多种多样，但其基本思路是相同的，即把被传输的信息源加在某一由混沌系统产生的混沌信号上，生成混合类噪声信号，对信号源加密，该信号发送到接收器上后，再由一相应的混沌系统分离其中的混沌信号，即解密过程，进而恢复出原输送的信息源。由于混沌同步效应的存在，使得这一解密过程能够实现 [3]。

温度测量

作者给出了一种基于映射式混沌电路的温度测量新方法，与现有的测量方法完全不同。把对温度敏感的所有元件参数都考虑在内，测量过程中将这些元件都置于测温环境中，电路输出的符号序列直接反映了温度的变化。因此不再有测量电路的温度漂移的问题，具有较高的精确度和灵敏度。此外，该电路的结构简单，又能直接输出反映温度变化的二进制代码，即数字信号，便于计算机处理。计算机仿真和实际电路实验都显示该方法的优越性 [4]。

附录 B

【代码页】

蔡氏电路数值模拟代码

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy.integrate import ode
4
5 def Chua(M1, M2, alpha, beta, axis):
6     # Define the ODE
7     h = lambda x: M2*x + (M1-M2)*(abs(x+1)-abs(x-1))/2
8     def derivative(t, v, d):
9         dxdt = alpha * (v[1] - v[0] - h(v[0]))
10        dydt = v[0] - v[1] + v[2]
11        dzdt = -beta * v[1]
12        return [dxdt, dydt, dzdt]
13
14     # ODE solver
15     v0 = [0.7, 0.0, 0.0]
16     t0 = 0.0
17     r = ode(derivative).set_integrator('dopri5')
18     r.set_initial_value(v0, t0).set_f_params(1)
19
20     # Time series
21     tmax = 50
22     dt = 0.02
23     num_steps = int(tmax / dt) + 2
24
25     # Generate the trace
26     x = np.zeros(num_steps - 1)
27     y = np.zeros(num_steps - 1)
28     z = np.zeros(num_steps - 1)
29     x[0], y[0], z[0] = v0
30     idx = 1
31
32     while r.successful() and r.t < tmax:
33         r.integrate(r.t+dt)
34         x[idx], y[idx], z[idx] = r.y[0:3]
35         idx += 1
36
37     # Plot
38     plt.figure(figsize=(9, 6))
39
40     if axis == "x-y":
41         plt.plot(x, y, color='lightseagreen',
42                   label='M1 = {:.1f}, M2 = {:.1f}, alpha = {:.1f}, beta = {:.1f}'.format(M1, M2, alpha, beta))
43
44     else:
45         plt.plot(y, z, color='lightseagreen',
46                   label='M1 = {:.1f}, M2 = {:.1f}, alpha = {:.1f}, beta = {:.1f}'.format(M1, M2, alpha, beta))
47
48     plt.xlabel('X')
49     plt.ylabel('Y' if axis == "x-y" else 'Z')
50     plt.title('Chua Circuit Phase Plane Plot')
51
52     # Set aspect ratio to 1
53     plt.gca().set_aspect('equal', 'box')
54
55     # Show plot
56     plt.show()

```

```
39         {:.1f}’
40         .format(M1, M2, alpha, beta))
41     plt.xlabel(‘x’)
42     plt.ylabel(‘y’)
43     plt.grid()
44     plt.legend(loc=‘center left’, bbox_to_anchor=(0.2, -0.15))
45 elif axis == “x-z”:
46     plt.plot(x, z, color=‘lightseagreen’,
47               label=‘M1 = {:.1f}, M2 = {:.1f}, alpha = {:.1f}, beta = {:.1f}’
48               .format(M1, M2, alpha, beta))
49     plt.xlabel(‘x’)
50     plt.ylabel(‘z’)
51     plt.grid()
52     plt.legend(loc=‘center left’, bbox_to_anchor=(0.2, -0.15))
53 elif axis == “y-z”:
54     plt.plot(y, z, color=‘lightseagreen’,
55               label=‘M1 = {:.1f}, M2 = {:.1f}, alpha = {:.1f}, beta = {:.1f}’
56               .format(M1, M2, alpha, beta))
57     plt.xlabel(‘y’)
58     plt.ylabel(‘z’)
59     plt.grid()
60     plt.legend(loc=‘center left’, bbox_to_anchor=(0.2, -0.15))
61 elif axis == “3d”:
62     fig = plt.figure(figsize=(9, 6))
63     ax = fig.add_subplot(projection=‘3d’)
64     plt.plot(x, y, z, color=‘lightseagreen’,
65               label=‘M1 = {:.1f}, M2 = {:.1f}, alpha = {:.1f},
66               beta = {:.1f}’
67               .format(M1, M2, alpha, beta))
68     ax.set_xlabel(‘x’)
69     ax.yaxis.set_label_text(‘y’)
70     ax.xaxis.set_label_text(‘z’)
71     plt.legend(loc=‘center left’, bbox_to_anchor=(0.2, -0.15))
72     plt.grid()
73 elif axis == “wave”:
74     fig, ax = plt.subplots(3, 1, figsize=(9, 6))
75     ax[0].plot(x, color=‘lightseagreen’)
76     ax[0].set_xlabel(‘Time t/s’)
77     ax[0].set_ylabel(‘x’)
78     ax[0].grid()
```

```

79     ax[1].set_xlabel('Time t/s')
80     ax[1].set_ylabel('y')
81     ax[1].grid()
82
83     ax[2].plot(z, color='lightseagreen')
84     ax[2].set_xlabel('Time t/s')
85     ax[2].set_ylabel('z')
86     ax[2].grid()
87     plt.subplots_adjust(left=None, bottom=None, right=None, top=None, wspace
88                         =0.2, hspace=0.2)
89
90 #line
91 for axis in ['x-y', 'y-z', 'x-z', '3d', 'wave']:
92     Chua(M1=-2.0, M2=-1.3, alpha=26, beta=12, axis=axis)
93
94 #limit circle
95 for axis in ['x-y', 'y-z', 'x-z', '3d', 'wave']:
96     Chua(M1=-0.5, M2=1.0, alpha=13, beta=13, axis=axis)
97
98 #double attractors
99 for axis in ['x-y', 'y-z', 'x-z', '3d', 'wave']:
100    Chua(M1=-1.1, M2=-0.7, alpha=21, beta=36, axis=axis)
101
102 #2st single attractor
103 for axis in ['x-y', 'y-z', 'x-z', '3d', 'wave']:
104    Chua(M1=-1.1, M2=-0.7, alpha=21, beta=31.3, axis=axis)
105
106 #2nd single attractor
107 for axis in ['x-y', 'y-z', 'x-z', '3d', 'wave']:
108    Chua(M1=-1.1, M2=-0.7, alpha=21, beta=150, axis=axis)

```

洛伦兹系统数值模拟代码

```

1 import numpy as np
2 from scipy import integrate
3 import matplotlib.pyplot as plt
4 from mpl_toolkits.mplot3d import Axes3D
5
6 def lorenz(p,t,s,r,b):
7     x,y,z = p.tolist()          #无质量点的当前位置(x,y,z)
8     print("x,y,z,t:",x,y,z,t)  #帮助理解odeint的执行过程
9     return s*(y-x),x*(r-z)-y,x*y-b*z #返回dx/dt,dy/dt,dz/dt
10

```

```
11 t = np.arange(0,100,0.01)
12 track = integrate.odeint(lorenz,(0.0,1.00,0.0),t,args=(10.0,28.0,2.6))
13 #track2 = integrate.odeint(lorenz,(0.0,1.01,0.0),t,args=(10.0,28.0,2.6))
14 #print("type(track):",type(track),"track.shape:",track.shape)

15
16 fig = plt.figure(figsize=(12,6))
17 ax = fig.gca(projection='3d')    #获取当前子图，指定三维模式
18 ax.plot(track[:,0],track[:,1],track[:,2],lw=1.0,color='r')  #画轨迹
19 ax.set_xlabel('x')
20 ax.set_ylabel('y')
21 ax.set_zlabel('z')
22 plt.show()

23
24 plt.plot(track[:,0], track[:,1])
25 plt.xlabel('x')
26 plt.ylabel('y')
27 plt.show()

28
29 plt.plot(track[:,0], track[:,2])
30 plt.xlabel('x')
31 plt.ylabel('z')
32 plt.show()

33
34 plt.plot(track[:,1], track[:,2])
35 plt.xlabel('y')
36 plt.ylabel('z')
37 plt.show()

38
39 fig2, axs = plt.subplots(3, sharex=True)
40 fig2.suptitle('x, y and z against t')
41 axs[0].plot(t, track1[:,0])
42 axs[1].plot(t, track1[:,1])
43 axs[2].plot(t, track1[:,2])

44
45 axs[0].set(ylabel = 'x')
46 axs[1].set(ylabel = 'y')
47 axs[2].set(ylabel = 'z')

48
49 axs[1].set(xlabel = 't')

50
51 axs[0].set_ylim(-20, 20)
52 axs[1].set_ylim(-25, 25)
53 axs[2].set_ylim(5, 50)
```

```
55 plt.xticks(np.arange(min(t), max(t)+1, 10))  
56 plt.show()
```