ГУАП

КАФЕДРА № 43

ОТЧЁТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| д-р техн. наук, доцент |  |  |  | С. И. Колесникова |
| должность, уч. Степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ОТЧЁТ О ЛАБОРАТОРНОЙ РАБОТЕ №4 | | | | | |
| ГРАФИЧЕСКАЯ ВИЗУАЛИЗАЦИЯ ФАЗОВЫХ ПОРТРЕТОВ СИСТЕМ ДЕТЕРМИНИРОВАННОГО ХАОСА | | | | | |
| по дисциплине: СИСТЕМНЫЙ АНАЛИЗ | | | | | |
|  | | | | | |
| РАБОТУ ВЫПОЛНИЛ | | | | | |
| СТУДЕНТ ГР. | 4330М |  |  |  | А.А. Кинько |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

**Цель работы**

Целью работы является освоение алгоритма построения бифуркационной диаграммы для дискретной модели.

**Текст задания**

Согласно варианту №8:

Создать exe-файл (для конечного пользователя) для решения задачи.

Ввод параметров модели – с клавиатуры.

Текст программы снабдить комментариями.

Дано непрерывное отображение Лоренца с эволюционными уравнениями и параметрами 



1. Построить зависимости  в условиях . .

2. Построить фазовые портреты  для случаев п.10 с разной степенью точности вычисления (2 знака в мантиссе, 15 знаков в мантиссе).

3. Привести примеры практического применения модели Лоренца.

4. Дать определение устойчивой и неустойчивой динамической системы.

**Ход работы**

В общем виде алгоритма построения бифуркационной диаграммы для дискретной модели имеет вид:

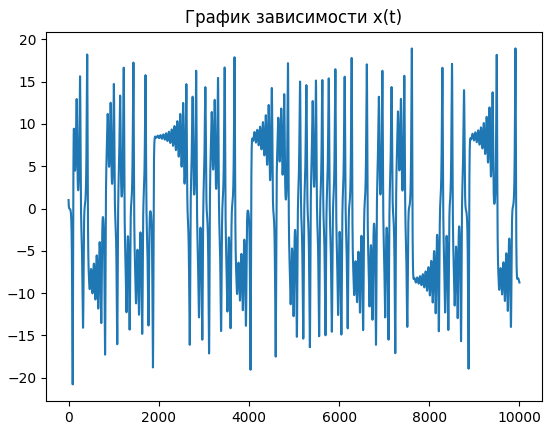
1. Выбираем начальное значение (например, ) и начальное значение параметра  (например, ).

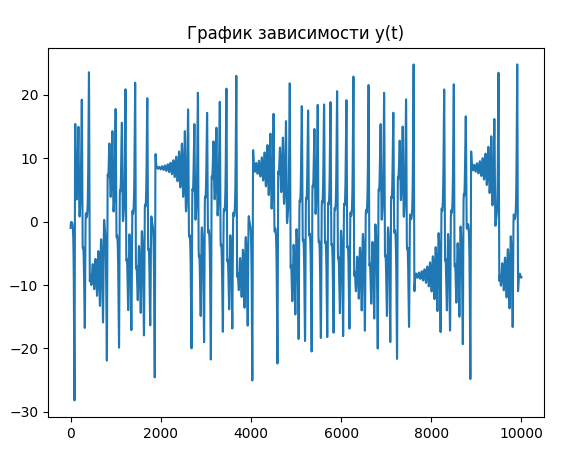
2. Производим 1000 (например) итераций заданного отображения с параметром  с целью достижения устойчивых предельных значений 

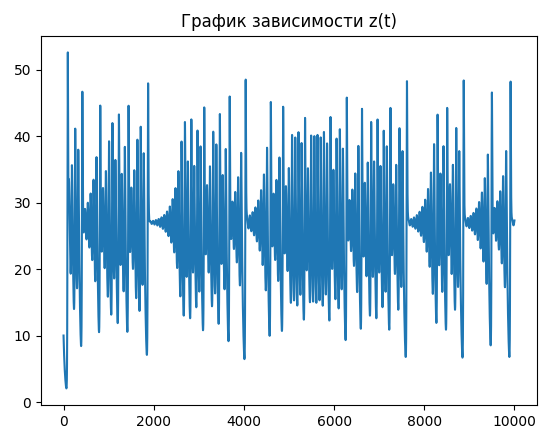
3. Запоминаем или отображаем (нанести на график по вертикальной оси) значения последних 200-300 итераций отображения. График на плоскости (λ*,x*).

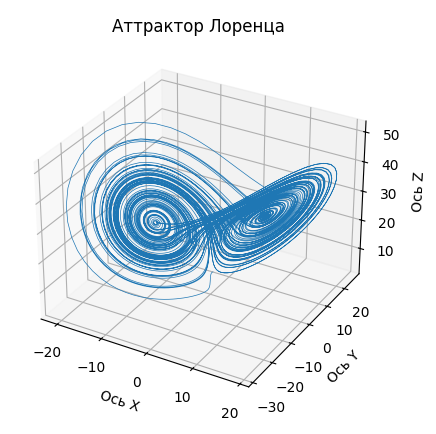
4. Увеличиваем значение на заданный шаг и повторяем процедуру вычислений до порогового значения, например .

Получим фазовые портреты для значений :

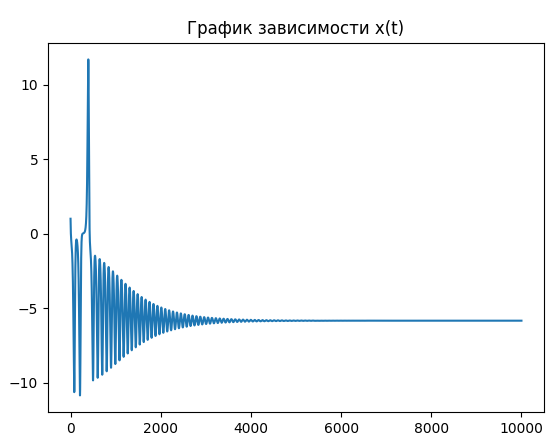


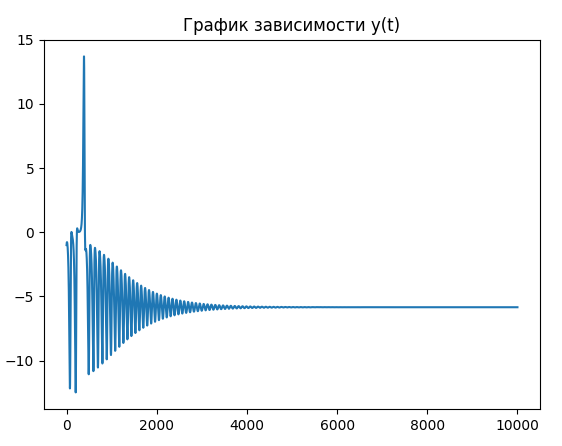


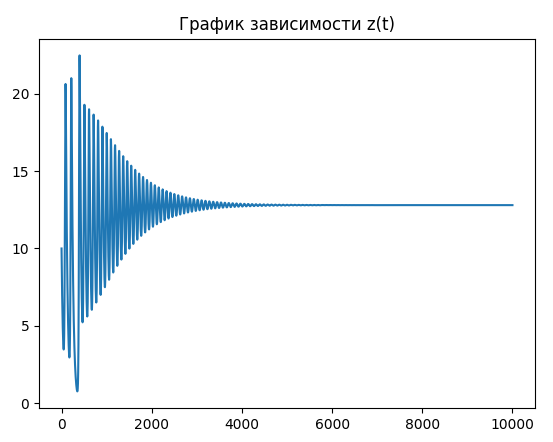


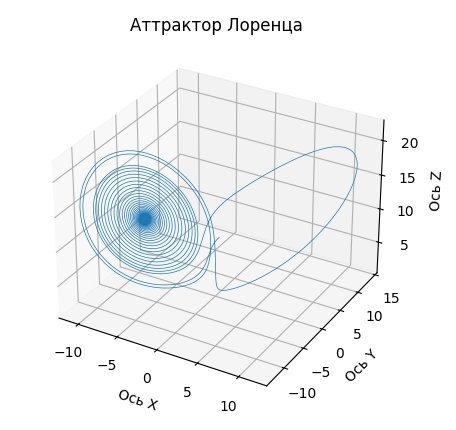


Как видно из рисунков, система неустойчива. Получен аттрактор Лоренца. Приведем систему к устойчивому состоянию, положив параметры :

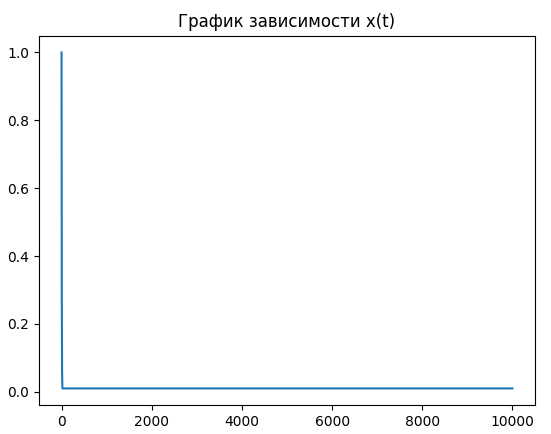


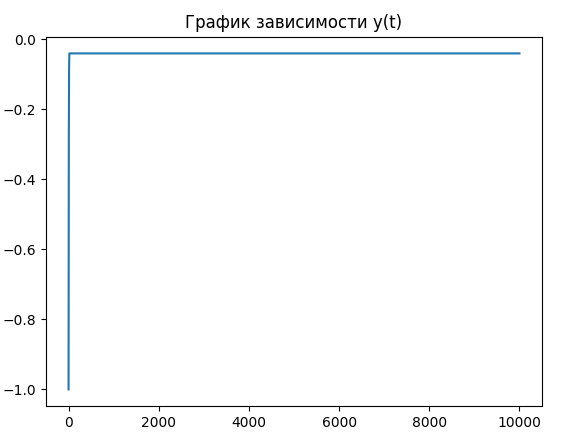


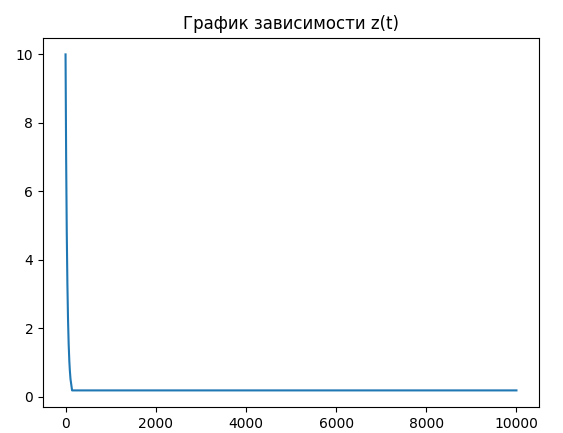


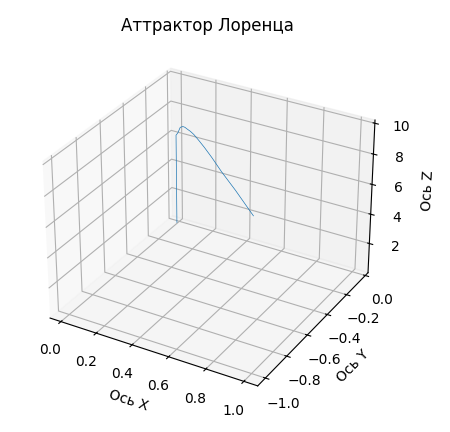


Предыдущие вычисления были выполнены в формате чисел с плавающей запятой с 15 знаками в мантиссе. Теперь вычислим те же значения в числах с мантиссой равной 2.









**Выводы**

В ходе выполнения лабораторной работы был реализован аттрактор Лоренца, а также было выведено, что он является неустойчивой системой при и устойчивой при . Также в зависимости от точности вычислений были получены разные результаты, свидетельствующие о значительной важности использования большей мантиссы для такого рода вычислений.

**Приложение А**

Листинг скрипта Python:

import matplotlib.pyplot as plt  
  
# Вариант №8  
  
# Дано непрерывное отображение Лоренца с эволюционными уравнениями и параметрами s, r, d:  
# x\*(t) = sy - sx;  
# y\*(t) = -y + rx - xz  
# z\*(t) = -bz + xy  
  
# 1. Построить зависимости x(t), y(t), z(t) в условиях:  
# [x, y, z] = [1, -1, 10]  
# s, r, b = 10, 28, 8/3  
# 2. Построить фазовые портреты (x, y, z) для случаев п.1^0 с разной степенью точности вычисления:  
# (2 или 15 знаков в мантиссе)  
  
def calculate\_lorenz(x\_start, y\_start, z\_start, s, r, b, iter\_num, dt, precision):  
 x = [x\_start]  
 y = [y\_start]  
 z = [z\_start]  
 current\_x = x\_start  
 current\_y = y\_start  
 current\_z = z\_start  
 for \_ in range(iter\_num):  
 x.append(round(current\_x + (s \* current\_y - s \* current\_x) \* dt, precision))  
 y.append(round(current\_y + (-current\_y + r \* current\_x - current\_x \* current\_z) \* dt, precision))  
 z.append(round(current\_z + (-b \* current\_z + current\_x \* current\_y) \* dt, precision))  
 current\_x = x[-1]  
 current\_y = y[-1]  
 current\_z = z[-1]  
  
 return x, y, z  
  
  
def show\_plots(x, y, z, iter\_num):  
 t = list(range(iter\_num + 1))  
 plt.plot(t, x)  
 plt.title("График зависимости x(t)")  
 plt.show()  
  
 plt.plot(t, y)  
 plt.title("График зависимости y(t)")  
 plt.show()  
  
 plt.plot(t, z)  
 plt.title("График зависимости z(t)")  
 plt.show()  
  
 ax = plt.figure().add\_subplot(projection='3d')  
  
 ax.plot(x, y, z, lw=0.5)  
 ax.set\_xlabel("Ось X")  
 ax.set\_ylabel("Ось Y")  
 ax.set\_zlabel("Ось Z")  
 ax.set\_title("Аттрактор Лоренца")  
 plt.show()  
  
  
x\_s, y\_s, z\_s = 1, -1, 10  
s\_s, r\_s, b\_s = 10, 28, 8/3  
cycles = 10000  
dt\_s = 0.01  
precision\_s = 15  
x\_data, y\_data, z\_data = calculate\_lorenz(x\_s, y\_s, z\_s, s\_s, r\_s, b\_s, cycles, dt\_s, precision\_s)  
print(x\_data)  
print(y\_data)  
print(z\_data)  
show\_plots(x\_data, y\_data, z\_data, cycles)