

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИМЕНИ М. В. ЛОМОНОСОВА  
ФИЛИАЛ МОСКОВСКОГО ГОСУДАРСТВЕННОГО  
УНИВЕРСИТЕТА ИМЕНИ М. В. ЛОМОНОСОВА В ГОРОДЕ  
СЕВАСТОПОЛЕ

Факультет «Компьютерной математики»  
Направление подготовки «Прикладная математика и информатика»  
01.03.02 (бакалавр)

**ОТЧЕТ**  
**по лабораторной работе №4**  
**«Построение цепно-рекуррентного множества для**  
**дифференциального уравнения на примере уравнения**  
**Дуффинга»**

Работу выполнил:  
студент группы ПМ-401  
Хаметов Марк Владимирович

Руководитель: профессор  
кафедры прикладной  
математики и информатики  
Осипенко Георгий Сергеевич

Севастополь, 2023

## Оглавление

Оглавление.....	2
Постановка задачи.....	3
Теоретическая часть.....	4
Интерфейс программы.....	6
Результаты.....	9
Использованная литература.....	10

## Постановка задачи

Дано уравнение Дуффинга:

$$\ddot{x} + k\dot{x} + \alpha x + \beta x^3 = B \cos(\omega t)$$

Произведем замену  $y = \dot{x}$  и получим динамическую систему уравнений:

$$\begin{cases} \dot{x} = y \\ \dot{y} = -k\dot{x} - \alpha x - \beta x^3 + B \cos(\omega t) \end{cases}$$

В области  $[-2; 2] \times [-2; 2]$  необходимо найти достаточно малую окрестность цепно-рекуррентного множества отображения этой динамической системы.

Для построения символического образа цепно-рекуррентного множества берется область. Эту область впоследствии разбивают на ячейки. Ячейки соответствующие цепно-рекуррентному множеству закрашиваются. Размер ячейки не превышает размера пикселя.

## Теоретическая часть

Сильно связанные вершины графа – это подмножества таких вершин ориентированного графа, между которыми существует путь в обоих направлениях.

Разбиение области на ячейки в данной работе - это разбиение на прямоугольники одинакового размера. Длина ребер задается пользователем. Нумерация ячеек идет в порядке сначала слева направо, затем сверху вниз. Тогда обозначим ячейку  $M(i)$ , где  $i$  номер вершины графа.

Тогда вершины графа это номера ячеек. Ребро исходящее из вершины соответствует отображению из соответствующей ячейки в другую ячейку. Номер полученной ячейки задает конечную точку ребра. Так как мы не можем отобразить каждую точку в области, мы отображаем  $k$  равномерно распределенных точек каждой ячейки. Это число задается пользователем.

Для реализации отображения этих точек мы используем метод Рунге-Кутты. Шаг алгоритма задается пользователем. Начальное время алгоритма равняется нулю, а конечное время равняется периоду функции в правой части уравнения. В случае уравнения Дуффинга период равен:

$$T = \frac{2*\pi}{\omega}$$

По теореме 5.1 из источника [1]: Пусть  $P(d)$  - это окрестность равная объединению всех ячеек соответствующих возвратным вершинам графа, где  $d$  - это длина стороны ячейки.

$$P(d) = \{\cup M(i), i - \text{возвратная}\}$$

Тогда цепно-рекуррентное множество совпадает с пересечением множеств  $P(d)$  по формуле:

$$Q = \bigcap_{d>0} P(d)$$

По теореме 5.2 из источника [1]: При уменьшении размера ячейки новая окрестность оказывается вложена в старую. Из этого следует то, что уменьшение диаметра ячеек приводит к меньшему размеру окрестности. Таким образом, последовательность окрестностей монотонно убывает и сходится к цепно-рекуррентному множеству по формуле:

$$\lim_{k \rightarrow \infty} P_k = \bigcap_k P_k = Q.$$

Для подсчета номера ячейки полученного после отображения точки области применяем формулу:

$$n = [(x - x_{\min}) / h_x] + 1 + [(y_{\min} - y) / h_y] * [(x_{\max} - x_{\min}) / h_x]$$

Для нахождения сильно связанных вершин графа была использована функция `strongly_connected_components` из библиотеки `networkx` для языка программирования `python`.

Так как разбиение на необходимое количество ячеек сразу приводит к применению сложного алгоритма к огромному количеству ячеек, мы используем алгоритм локализации цепно-рекуррентного множества. Для этого мы разбиваем область на малое количество ячеек и ищем сильно связанные вершины графа. Затем мы отбрасываем остальные вершины, для этого мы создаем список ячеек с которыми необходимо продолжить работу. Мы разбиваем область на ячейки размером меньше. Далее мы применяем алгоритм только для ячеек из списка. Алгоритм продолжает работу до момента достижения желаемого размера ячеек.

# Интерфейс программы

Построение цепно-рекуррентного множества

— □ ×

Меню

Система уравнений

Значения параметров

Точки задающие область

Козфициент переразбиения и количество точек

Построить итераций и Достроить итераций

Построить график для итераций:

Проитерировать существующий:

Отчистить старые данные и занести новые

Меню



## Меню

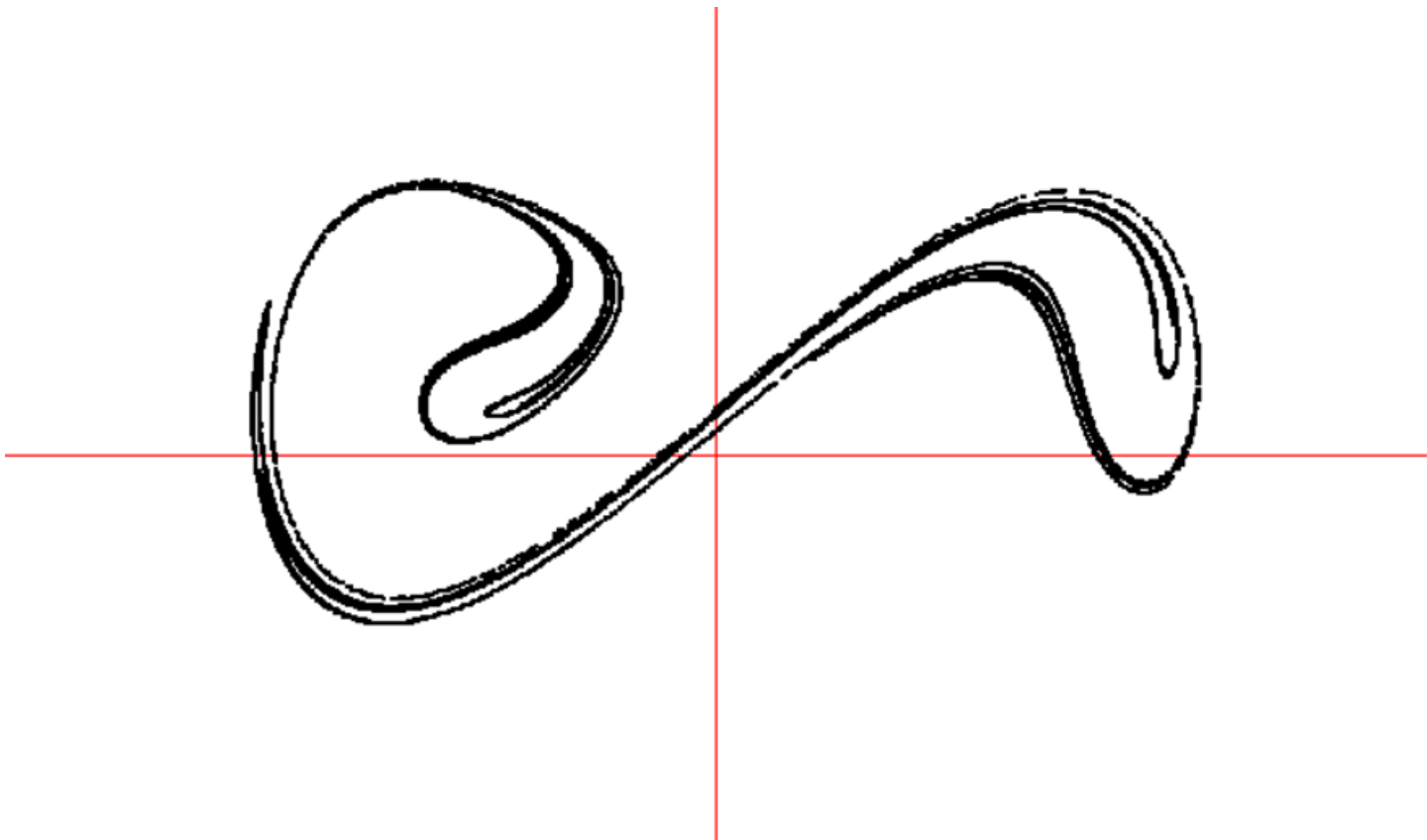
1 итерация. Занято времени 0.016015052795410156  
Количество ячеек 256.0  
2 итерация. Занято времени 0.04601025581359863  
Количество ячеек 1024.0  
3 итерация. Занято времени 0.17910289764404297  
Количество ячеек 4096.0  
4 итерация. Занято времени 0.7141475677490234  
Количество ячеек 16384.0  
5 итерация. Занято времени 2.876002788543701  
Количество ячеек 65536.0  
6 итерация. Занято времени 11.95172905921936  
Количество ячеек 262144.0 На этой итерации также был нарисован график  
7 итерация. Занято времени 48.47069954872131  
Количество ячеек 1048576.0  
8 итерация. Занято времени 210.70473718643188  
Количество ячеек 4194304.0 На этой итерации также был нарисован график



## Результаты

Программа написана на языке Python. Для создания интерфейса использовалась библиотека PySide6. Для расчетов использовались библиотеки networkx, math, time.

Для измерений результатов область задавалась  $[-2, -2] \times [2, 2]$  и на каждой итерации длина ребра ячейки делилась на два, изначальная длина ребра была равна 0.5 .



Для параметров  $\alpha = -1$  ,  $\beta = 1$  ,  $k = 0.25$  ,  $B = 0.3$  ,  $w = 1$  . Для 4 отображаемых точек в виде сетки 2 на 2 из ячейки. Для 62 шагов алгоритма Рунге-Кутты каждой точки. Восьмая итерация заняла 3,5 минуты. Всего программа исполнялась 4,5 минуты. Количество ячеек на финальном изображении равнялось 4194304.

## Использованная литература

1. “Введение в символический анализ динамических систем” Г.С.Осипенко, Н.Б.Ампилова.
2. [https://ru.wikipedia.org/wiki/Компонента\\_сильной\\_связности](https://ru.wikipedia.org/wiki/Компонента_сильной_связности)
3. [https://cyberleninka.ru/article/n/otsenka-pokazateley-lyapunova-metodamisimvolic\\_heskogo-analiza](https://cyberleninka.ru/article/n/otsenka-pokazateley-lyapunova-metodamisimvolic_heskogo-analiza)
4. <https://zetcode.com/gui/pysidetutorial/drawing/>
5. <https://github.com/Zenoro/ODU-solutions>
6. <https://www.freecodecamp.org/news/lambda-expressions-in-python/>
7. <https://www.geeksforgeeks.org/topological-sorting/>
8. <https://stackoverflow.com/questions/17200117/how-to-get-the-object-name-from-within-the-class>
9. <https://srinikom.github.io/pyside-docs/PySide/QtCore/QRectF.html>
10. [https://en.wikipedia.org/wiki/Coordinate\\_system](https://en.wikipedia.org/wiki/Coordinate_system)
11. <https://stackoverflow.com/questions/60918473/how-do-i-convert-pixel-screen-coordinates-to-cartesian-coordinates>
12. <https://www.pythonguis.com/tutorials/pyside6-plotting-pyqtgraph/>
13. <https://stackoverflow.com/questions/17200117/how-to-get-the-object-name-from-within-the-class>
14. <https://eltehhhelp.xyz/wp-content/uploads/2021/09/image-2.png>
15. <https://eltehhhelp.xyz/wp-content/uploads/2021/09/image-1.png>

