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)

# ОТЧЁТ
**по вычислительной задаче №9**
**«Вычисление спектра усреднения функции**
**для дискретной динамической системы (отображение Жюлиа)»**

Работу выполнил:
Студент группы ПМ-401
Воронец Владимир Олегович

Руководитель: профессор
кафедры прикладной
математики и информатики
Осипенко Георгий Сергеевич

Севастополь, 2023

## ОГЛАВЛЕНИЕ

## ПОСТАНОВКА ЗАДАЧИ

Вычислить спектр усреднения функции $\varphi(\mathrm{x}, \mathrm{y})$ над цепно-рекуррентным множеством дискретной системы (отображения Жюлиа):

$$x \to x^2 - y^2 + a$$

$$y \to 2xy + b$$

в области $R^2$: $[-2;\ 2]x[-2;\ 2]$ для функции $\varphi(\mathrm{x}, \mathrm{y}) = \mathrm{x}^2 + \mathrm{y}^2$

## ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Рассмотрим дискретную динамическую систему, порожденную гомеоморфизмом $f: M \to M$ компактного многообразия $M$ $x_{n+1} = f(x_n)$. Усреднением функции $\varphi: M \to \mathbb{R}$ над конечной последовательностью точек $\chi = \{x_k, 0 < k \leq n\}$ называется среднее арифметическое значений функции в заданных точках:

$$\lambda(\chi) = \frac{1}{n}\sum_{k=1}^{n} \varphi(x_k).$$

**Теорема 1.** Спектр $\Sigma$ усреднения функции $\varphi$ состоит из отрезков $[a_k, b_k]$. Каждый отрезок генерируется компонентой цепно-рекуррентного множества $\Omega_k$, где $a_k = \lambda_{inf}(\Omega_k)$ и $b_k = \lambda_{sup}(\Omega_k)$ нижняя и верхняя грани усреднений, реализуемых на компоненте $\Omega_k$. [1]

Рассмотрим граф $G$, из каждой вершины которого исходит хотя бы одна дуга $e = (i \to j)$. Заметим, что описанный граф всегда имеет замкнутые пути, так как можно построить бесконечный путь, двигаясь вперед, в то время как число вершин конечно. Поэтому вершины на таком пути повторяются, что даёт замкнутые пути.

**Теорема 2. 1.** Спектр $\Sigma$ усреднения функции $\varphi$ лежит в расширенном спектре символического образа $\Sigma^*$, который состоит из интервалов $[\lambda_{\min}(\mathrm{H_k})\text{-}\theta(\mathrm{d}), \lambda_{\max}(\mathrm{H_k}) + \theta(\mathrm{d})]$, где $\{H_\mathrm{k}\}$ есть полное семейство классов эквивалентных возвратных вершин символического образа, $d$ – диаметр покрытия и $\theta(d)$ – модуль непрерывности функции $\varphi$.

**2.** Если диаметр покрытия $d \to 0$, то расширенный спектр $\Sigma^*$ и спектр символического образа $\Sigma(G)$ сходятся к $\Sigma$ в метрике Хауздорфа.

Для этого может быть использован алгоритм Романовского, который находит цикл с минимальным средним значением в компоненте сильной связности. Незначительно изменив его, можно найти и цикл с максимальным значением.

## РЕШЕНИЕ ПОСТАВЛЕННОЙ ЗАДАЧИ

Имеется граф символического образа динамической системы $G = (M, N)$, где M – множество вершин, N-множество ориентированных дуг.

Подготовительный шаг алгоритма Романовского – поиск базисного цикла.

1) Для каждой вершины $i \in M$ среди выходящих из $i$ дуг выбрать дугу $j(i)$ с наименьшим $c(j)$. Множество отобранных дуг обозначим через $N^*$. Подсчитать для каждой вершины $i$ ее степень $s[i]$ — число входящих в нее отобранных дуг.

2) Составить список вершин $M^*$ с нулевой степенью.

3) Пока список не пуст, выполнять следующие действия.

3.1) Исключить из списка $M^*$ первую вершину $i_1$.

3.2) Исключить ребро $j(i_1)$ из $N^*$.

3.3) Уменьшить на единицу степень вершины $i_2 = e(j)$. Если эта степень стала нулевой, то включить $i_2$ в список $M^*$.

4) Если список $M^* = \emptyset$, то дуги, входящие в $N^*$, образуют изолированные циклы. Полученные циклы нужно просмотреть и выбрать цикл с минимальным средним.

Далее выполняется непосредственно цикл самого алгоритма, и строится дерево, которое в итоге должно превратиться в экстремальный цикл:

1) Вычислить среднее значение $z$ на цикле $(M_c, N_c)$ и положить $v(i_0) = 0$.

2) Для всех дуг $j \in N_c$ вычислить потенциал

$$v\big(e(j)\big) = c(j) + v\big(b(j)\big) - z$$

в порядке следования вершин на цикле.

3) Удалить дугу $j$ цикла, входящую в $i_0$ и положить все вершины $M_c$ в список $M_1$.

4) Пока список $M_1$ не пуст выполнить:

4.1) Исключить из $M_1$ первую вершину $i_1$ и положить $M_0 := M_0 \cup i_1$

4.2) Для всех дуг $j = (i_1 \rightarrow i_2)$ с началом в $i_1$ вычислить

$$w = v(i_1) + c(j) - z.$$

4.2.1) Если $i_2 \in M_2$, то считаем $v(i_2) = w$, добавить к дереву дугу $j$ и переместить вершину $i_2$ из $M_2$ в конец списка $M_1$.

4.2.2) Если $i_2 \in M_0 \cup M_1$ и $w \geq v(i_2)$, то закончить обработку дуги.

4.2.3) Если $i_2 \in M_0 \cup M_1$ и $w < v(i_2)$, то возможны два варианта:

a) Если $i_2$ предшествует $i_1$ в построенном дереве, то найденный путь замыкает цикл со средним значением меньшим чем $z$. Выберем этот цикл как базисный и повторим алгоритм заново.

b) Если $i_2$ не предшествует $i_1$ в построенном дереве, то включить дугу $j$ в дерево, удаляя какое-либо ребро, входящее в $i_2$. Если при этом $i_2 \in M_0$, то переместить эту вершину из $M_0$ в начало списка $M_1$.

5) Если вычисления дошли до этого пункта, то построено дерево и потенциал, которые удовлетворяют требуемым неравенствам и базисный цикл реализует минимальное среднее значение оснащения.

# КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ

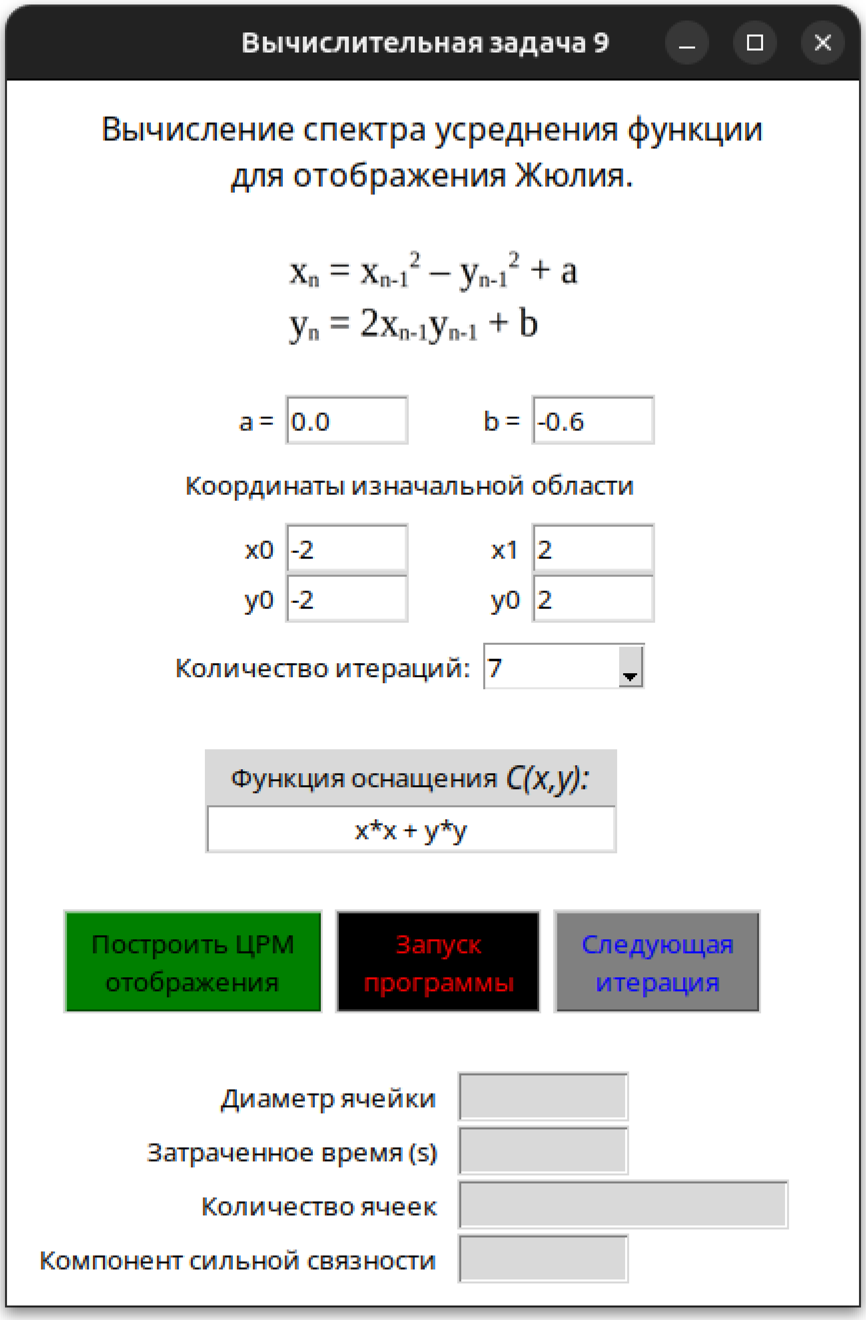


*Рисунок 1: Пользовательский интерфейс программы*

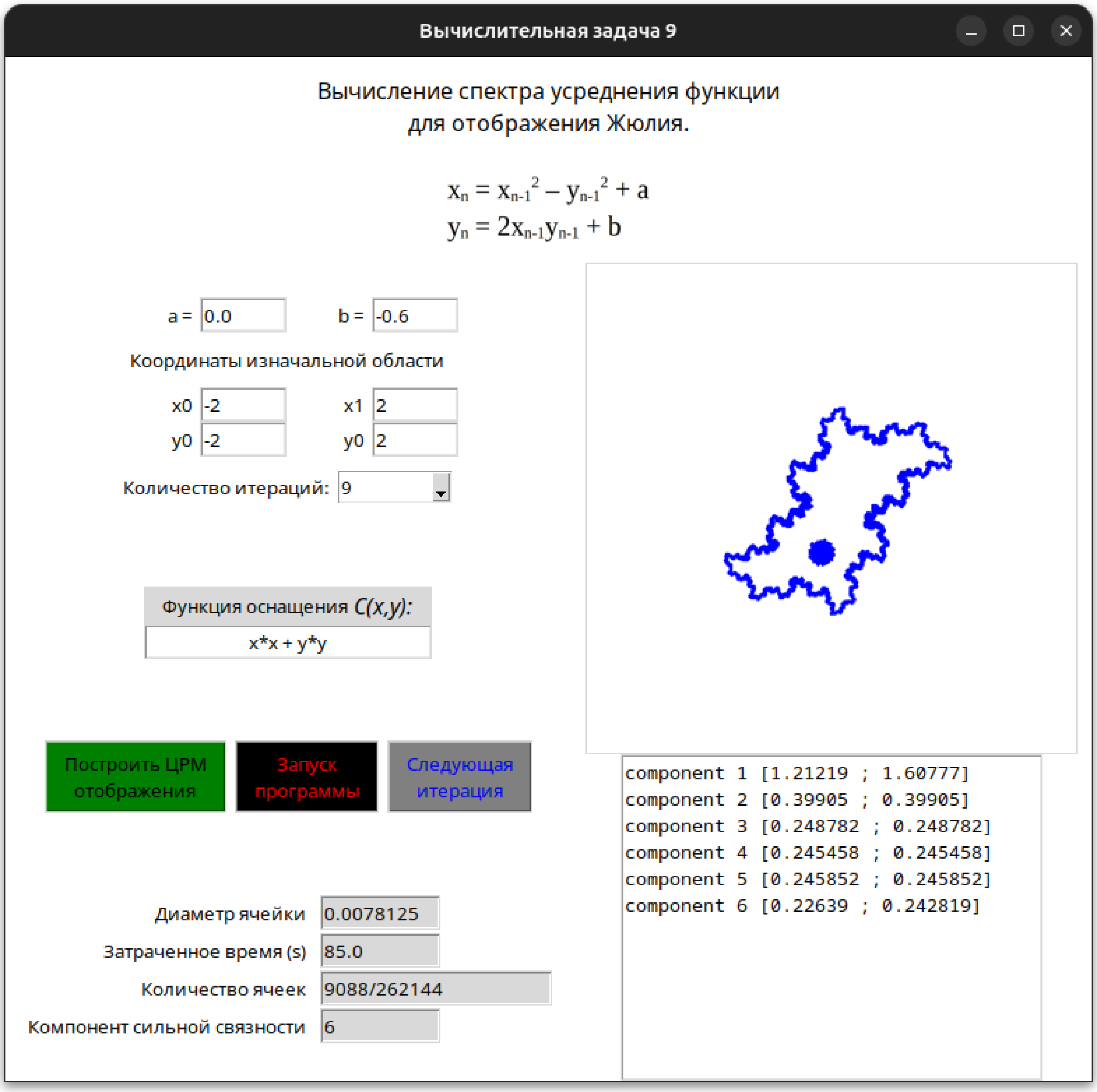


*Рисунок 2: Полученный результат при 9 итерациях и значениях параметров $a = 0$, $b = -0.6$ с заполненным информационным полем.*

# ХАРАКТЕРИСТИКА ПРОГРАММЫ

Время выполнения программы зависит от количества итераций: 9 итераций выполнено за 85 секунд, 7 итераций – за 6 секунд.

Было использовано 89 мегабайт памяти компьютера при подсчете ячеек и около 50 мегабайт при необязательном построении цепно-рекуррентного множества.

Нагрузка на процессор (AMD Ryzen 3 3200U) доходила до 83% при подсчете ячеек и ~50% при графическом отображении.

Программа была написана на языках программирования C++ для выполнения основного алгоритма и Python3 [2] с использованием графической библиотеки Matplotlib [3] для построения цепно-рекуррентного множества отображения ячеек, и библиотеки для создания оконных приложений Tkinter [4]. Программа ориентирована на UNIX-подобные системы. Необходимо предварительно установить все вышеперечисленные Python библиотеки.

# СПИСОК ЛИТЕРАТУРЫ

1. Осипенко Г.С. Компьютерно-ориентированные методы динамических систем: учебное пособие. – М: ИНФРА-М, 2023, – 295 с.
2. https://www.python.org/doc/
3. https://matplotlib.org/
4. https://docs.python.org/3/library/tkinter.html#module-tkinter