

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 №5**  
**«Построение цепно-рекуррентного множества**  
**в проективном пространстве»**

Работу выполнил:  
Студент группы ПМ-401  
Воронец Владимир Олегович

Руководитель: профессор  
кафедры прикладной  
математики и информатики  
Осипенко Георгий Сергеевич

Севастополь, 2023

## ОГЛАВЛЕНИЕ

<b>ПОСТАНОВКА ЗАДАЧИ .....</b>	<b>3</b>
<b>ТЕОРЕТИЧЕСКАЯ ЧАСТЬ.....</b>	<b>3</b>
<b>РЕШЕНИЕ ПОСТАВЛЕННОЙ ЗАДАЧИ.....</b>	<b>4</b>
<b>КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ.....</b>	<b>5</b>
<b>ХАРАКТЕРИСТИКА ПРОГРАММЫ .....</b>	<b>9</b>
<b>СПИСОК ЛИТЕРАТУРЫ.....</b>	<b>9</b>

## ПОСТАНОВКА ЗАДАЧИ

Построить цепно-рекуррентное множество в трехмерном проективном пространстве для трех различных случаев матрицы оператора линейного преобразования.

## ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Пусть  $R^{n+1}$  – векторное пространство размерности  $n + 1$  над полем  $R$  вещественных чисел.  $V$  – множество векторов данного пространства. Непустое множество  $P$  называется проективным пространством  $n$  размерности, порожденным пространством  $R^{n+1}$ , если задано отображение  $A: V/\{0\} \rightarrow P$ , удовлетворяющие 2 аксиомам проективного пространства:

1.  $A$ -сюръективное отображение.
2.  $A(x) = A(y)$  тогда и только когда  $x$  и  $y$  коллинеарны.

Рассмотрим проективное пространство, порожденное трехмерным векторным пространством. На нем можно задать динамическую систему, порожденную оператором линейного преобразования  $A$ , который представим в виде матрицы, размерности  $3 \times 3$ . Само преобразование осуществляется по формуле:  $e_{n+1} = Ae_n$

При этом каждую, кроме нулевой, точку линейного пространства можно отобразить на соответствующую точку проективного пространства, выполнив нормировку, разделив все компоненты вектора на максимальное значение компоненты в векторе.

В нашем случае проективное пространство представимо в виде 3 квадратов, так называемых локальных карт размерном  $[-1; 1] \times [-1; 1]$  в осях  $(oY, oZ); (oX, oZ); (oX, oY)$ . И можно отобразить точку на соответствующей карте при  $x = 1, y = 1, z = 1$  в нормированном векторе. [1]

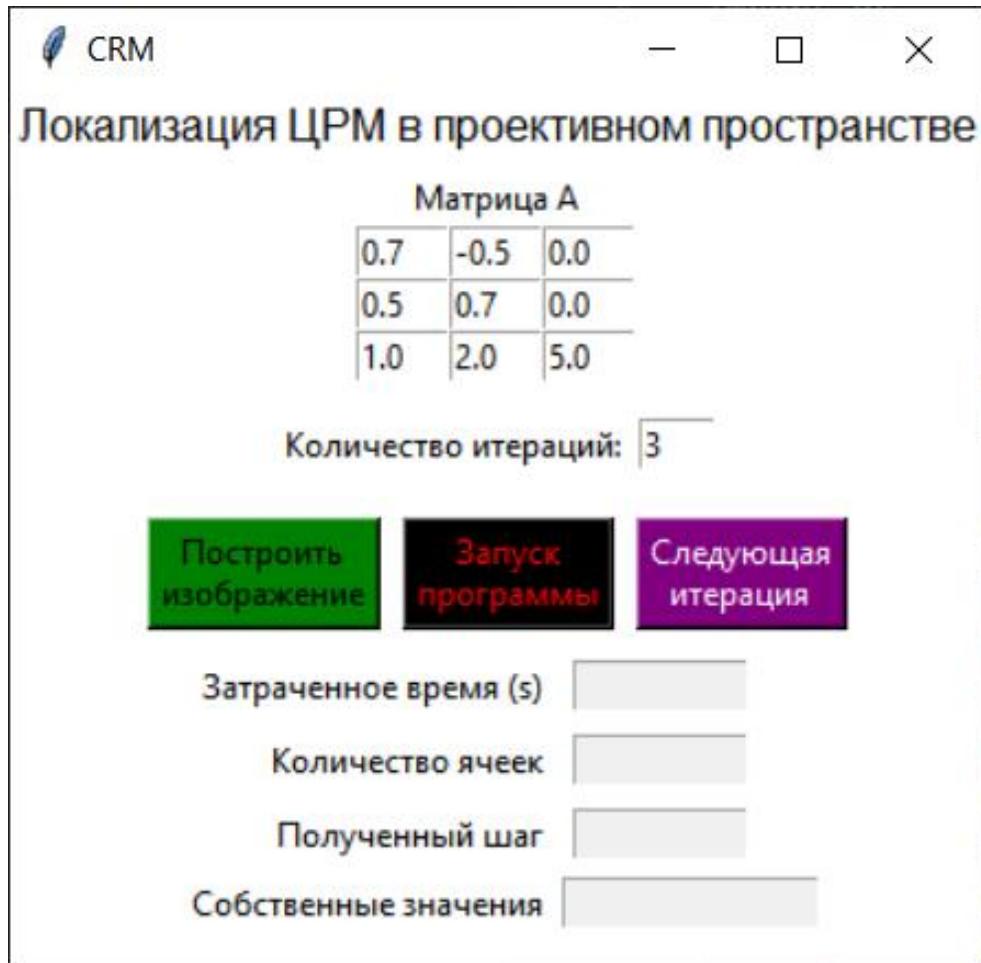
## РЕШЕНИЕ ПОСТАВЛЕННОЙ ЗАДАЧИ

Чтобы построить цепно-рекуррентное множество для данной динамической системы, необходимо построить символический образ системы, а затем выделить в получившемся графе компоненты сильной связности. Те компоненты, в которые входит по одной вершине, далее не учитываются. Все остальные вершины (входящие в остальные компоненты) являются возвратными и учитываются. Совокупность ячеек, соответствующих данным вершинам, составит окрестность искомого цепно-рекуррентного множества. Чем меньше диаметр ячеек, тем меньше окрестность и тем точнее приближается цепно-рекуррентное множество. Предлагается вначале построить цепно-рекуррентное множество для начального разбиения на 4 ячейки для каждой локальной карты, а затем дробить диаметр пополам  $n$  раз и повторять все процедуры, за исключением того, что ячейки, на каком-то этапе не попавшие в окрестность цепно-рекуррентного множества, на всех последующих этапах не рассматриваются.

Отображение локализации цепно-рекуррентного множества в проективном пространстве будет зависеть от собственных чисел оператора преобразования в этом пространстве, порождающего динамическую систему, так как

$Av = \lambda v$ , и порождается возвратная вершина.

## КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ



CRM

Локализация ЦРМ в проективном пространстве

Матрица A

0.7	-0.5	0.0
0.5	0.7	0.0
1.0	2.0	5.0

Количество итераций:

**Построить изображение** **Запуск программы** **Следующая итерация**

Затраченное время (s)

Количество ячеек

Полученный шаг

Собственные значения

Рисунок 1: Пользовательский интерфейс программы

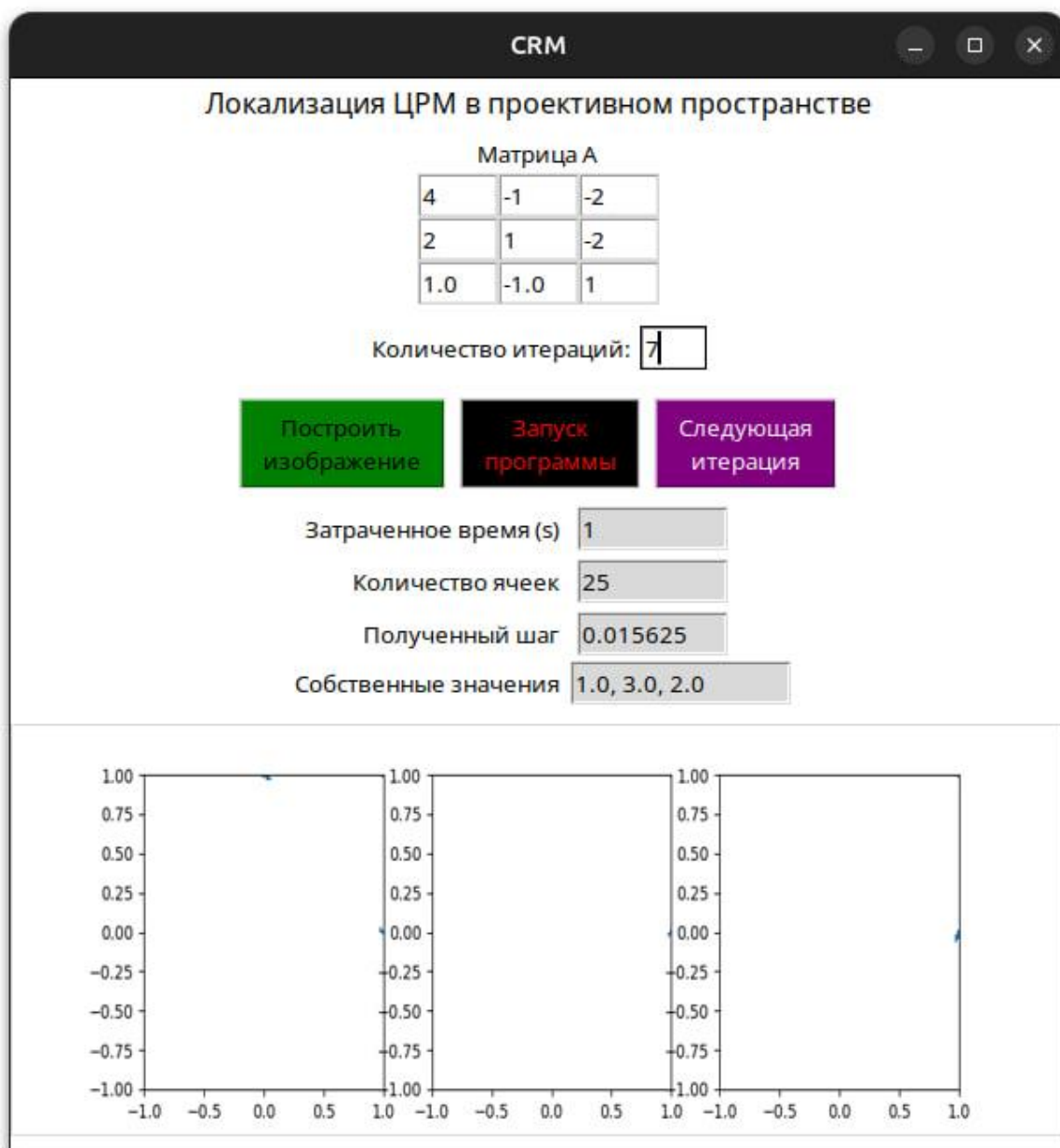
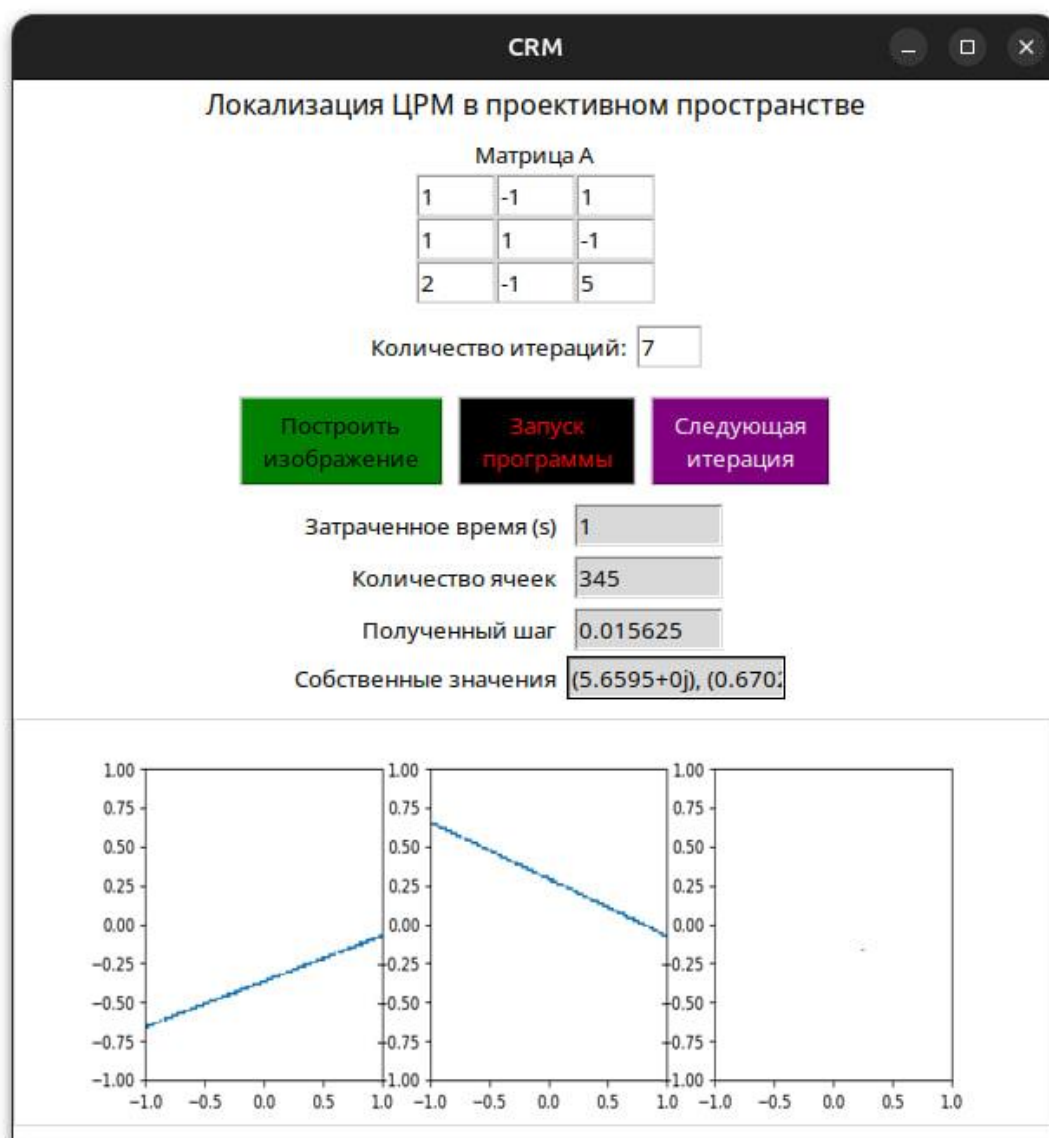


Рисунок 2: Цепно-рекуррентное множество для матрицы с тремя различными действительными собственными значениями (1, 2, 3)

В данном примере собственные значения матрицы – три различных действительных числа. Конечное разбиение  $d$  достигло значения 0.015625, отображено 25 ячеек в виде двух точек: первая смещена положительно относительно  $oY$  на первой карте, вторая находится в центре локальной карты в осях  $(oX, oY)$ .



*Рисунок 3: Цепно-рекуррентное множество для матрицы с тремя различными собственными числами (вкл. мнимое).*

В данном примере собственные значения матрицы – три различных числа, содержащие мнимую часть. Конечное разбиение  $d$  достигло значения 0.015625, отображено 345 ячеек в виде двух прямых на осях  $(oY, oZ)$  и  $(oX, oZ)$ , и неподвижной точки на третьей карте. Комплексные собственные числа матрицы динамического оператора порождают не только неподвижные точки, но и прямые, которые входят в цепно-рекуррентное множество в проективном пространстве.

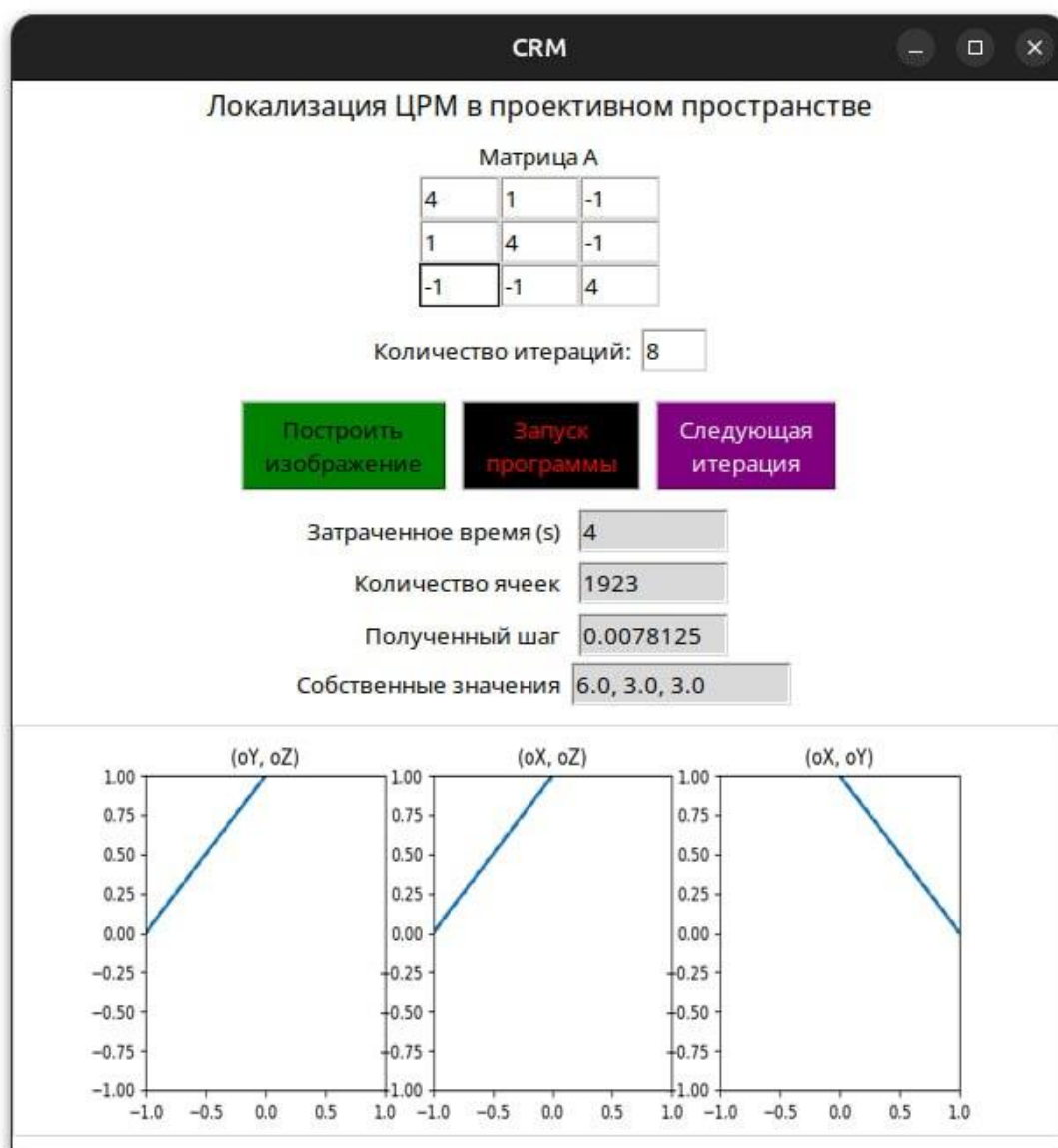


Рисунок 4: Цепно-рекуррентное множество для матрицы с двумя действительными собственными числами.

В данном примере собственные значения матрицы – два различных действительных числа. Конечное разбиение  $d$  достигло значения 0.0078125, отображено 1923 ячейки в виде трех прямых на трех картах.



## ХАРАКТЕРИСТИКА ПРОГРАММЫ

Время выполнения программы в среднем составляло около 0-4 секунды для подсчета ячеек и около 3-8 секунд для графического отображения.

Было использовано 121 мегабайт памяти компьютера при подсчете ячеек и около 80 мегабайт при графическом построении результата.

Нагрузка на процессор (AMD Ryzen 3 3200U) доходила до 90% при подсчете ячеек и около 40% при графическом отображении.

Программа была написана самостоятельно на двух языках программирования: Python3 [2] с использованием графической библиотеки Matplotlib [3] для графического отображения ячеек и библиотеки для создания оконных приложений Tkinter [4]; C++ использовался для выполнения операций алгоритма решения, что помогает нагрузить процессор на максимум и выполнять подсчет ячеек быстрее. Программа ориентирована на UNIX-подобные системы. Необходимо предварительно установить все вышеперечисленные Python библиотеки.

## СПИСОК ЛИТЕРАТУРЫ

1. Осипенко Г.С., Ампилова Н.Б. Введение в символический анализ динамических систем: – СПб.: Издательство Санкт-Петербургского университета, 2004. 240 с.
2. <https://www.python.org/doc/>
3. <https://matplotlib.org/>
4. <https://docs.python.org/3/library/tkinter.html#module-tkinter>