Визуализация фракталов: Множества Мандельброта, Жюлиа и Бассейны Ньютона

Содержание

1 Введение **3**

2 Определения **3**
- 2.1 Множество Мандельброта . . . 3
- 2.2 Множество Жюлиа . . . 3

3 Математические концепции **3**
- 3.1 Итерация и композиция . . . 3
- 3.2 Устойчивость итераций . . . 3

4 Доказательства свойств множества Мандельброта **4**
- 4.1 Доказательство симметрии относительно вещественной оси . . . 4
- 4.2 Доказательство ограничения по модулю . . . 4

5 Визуализация множества Мандельброта **5**
- 5.1 Описание алгоритма . . . 5
- 5.2 Реализация на Python . . . 5
- 5.3 Результаты визуализации . . . 6

6 Визуализация множества Жюлиа **7**
- 6.1 Описание алгоритма . . . 7
- 6.2 Реализация на Python . . . 7
- 6.3 Результаты визуализации . . . 8

7 Исследование другого фрактала: Фрактал Виксека **9**
- 7.1 Описание фрактала . . . 9
- 7.2 Алгоритм построения . . . 9
- 7.3 Реализация на Python . . . 9
- 7.4 Результаты визуализации . . . 10
- 7.5 Анализ результатов . . . 11

8 Набор изображений при разных параметрах **12**
- 8.1 Множество Мандельброта при разных итерациях . . . 12
- 8.2 Приближение отдельных частей множества Мандельброта . . . 13
- 8.3 Множество Жюлиа при разных значениях c . . . 13

9 Заключение **14**

1 Введение

В этой лабораторной работе мы исследуем и визуализируем фрактальные множества: множество Мандельброта, множества Жюлиа и бассейны Ньютона. Мы изучим их математические свойства, реализуем алгоритмы для генерации их визуальных представлений и проанализируем полученные результаты.

2 Определения

2.1 Множество Мандельброта

Определение: Множество Мандельброта определяется как множество комплексных чисел c, для которых последовательность, заданная уравнением

$$z_{n+1} = z_n^2 + c, \quad z_0 = 0, \tag{1}$$

остаётся ограниченной при $n \to \infty$. Иными словами, если величина $|z_n|$ не стремится к бесконечности для данного c, то c является частью множества Мандельброта.

2.2 Множество Жюлиа

Определение: Для заданного комплексного параметра c множество Жюлиа J_c — это множество всех точек z_0 на комплексной плоскости, для которых последовательность

$$z_{n+1} = z_n^2 + c, \tag{2}$$

начиная с z_0, остаётся ограниченной. Множество Жюлиа можно рассматривать как границу между точками, которые приводят к ограниченным и неограниченным последовательностям при итерации функции.

3 Математические концепции

3.1 Итерация и композиция

Итерация функции: Обозначение $f^n(z)$ представляет собой n-кратную итерацию функции f, то есть повторное применение функции f к её результату:

$$f^n(z) = \underbrace{f \circ f \circ \cdots \circ f}_{n \text{ раз}}(z). \tag{3}$$

Здесь символ $\circ$ обозначает композицию функций, а именно $f \circ g(z) = f(g(z))$.

3.2 Устойчивость итераций

Устойчивость итераций: Итерации функции f считаются устойчивыми в точке z_0, если небольшие изменения в начальном значении z_0 не приводят к значительным изменениям в последовательности $f^n(z_0)$. Математически это выражается следующим образом:

$$\forall \varepsilon > 0 \; \exists \delta > 0 : \; \forall z, \; |z - z_0| < \delta \implies |f^n(z) - f^n(z_0)| < \varepsilon, \; \forall n \in \mathbb{N}. \tag{4}$$

Это означает, что траектории итераций точек, близких к z_0, остаются близкими на всех шагах итерации.

4 Доказательства свойств множества Мандельброта

4.1 Доказательство симметрии относительно вещественной оси

Свойство 1: Множество Мандельброта симметрично относительно вещественной оси, то есть если c принадлежит множеству, то и его комплексно-сопряжённое $\overline{c}$ также принадлежит множеству.

Доказательство: Рассмотрим последовательность:

$$z_{n+1} = z_n^2 + c, \quad z_0 = 0. \tag{5}$$

Пусть для некоторого c последовательность $\{z_n\}$ ограничена. Рассмотрим комплексно-сопряжённое число $\overline{c}$ и соответствующую последовательность $\{\overline{z}_n\}$. Покажем, что она также ограничена.

Используем свойство комплексного сопряжения:

$$\overline{z_{n+1}} = \overline{z_n^2 + c} = \overline{z_n}^2 + \overline{c}. \tag{6}$$

Это означает, что последовательность для $\overline{c}$ является комплексно-сопряжённой к последовательности для c. Поскольку модуль комплексно-сопряжённого числа равен модулю исходного числа, то $|\overline{z}_n| = |z_n|$. Таким образом, если последовательность $\{z_n\}$ ограничена, то и $\{\overline{z}_n\}$ ограничена.

Следовательно, если c принадлежит множеству Мандельброта, то и $\overline{c}$ также принадлежит множеству. Это доказывает симметрию множества Мандельброта относительно вещественной оси.

4.2 Доказательство ограничения по модулю

Свойство 2: Если $|c| > 2$, то комплексное число c не принадлежит множеству Мандельброта.

Доказательство: Начнём с $z_0 = 0$. Тогда $z_1 = c$. Если $|c| > 2$, то $|z_1| > 2$. Рассмотрим следующий шаг:

$$|z_{n+1}| = |z_n^2 + c| \geq |z_n|^2 - |c|. \tag{7}$$

Поскольку $|z_n| > 2$ и $|c| > 2$, имеем:

$$|z_{n+1}| \geq |z_n|^2 - |c| > 2^2 - 2 = 2. \tag{8}$$

Это означает, что модуль $|z_n|$ будет расти и стремиться к бесконечности при $n \to \infty$. Таким образом, последовательность неограничена, и c не принадлежит множеству Мандельброта.

5 Визуализация множества Мандельброта

5.1 Описание алгоритма

Для визуализации множества Мандельброта используем следующий алгоритм:

1. **Определение области комплексной плоскости:** Задаём диапазоны по осям x (действительная часть) и y (мнимая часть), например, $x \in [-2.5, 1]$, $y \in [-1, 1]$. Создаём сетку точек с заданным разрешением.

2. **Итерационный процесс:** Для каждой точки $c = x + iy$ выполняем итерации:

$$z_{n+1} = z_n^2 + c, \quad z_0 = 0, \tag{9}$$

   до достижения максимального числа итераций или пока $|z_n| > 2$.

3. **Определение цвета пикселя:** Если последовательность не выходит за пределы круга радиуса 2 после максимального числа итераций, то точка принадлежит множеству и окрашивается в чёрный цвет. Иначе пиксель окрашивается в цвет, зависящий от числа итераций, потребовавшихся для "убегания"последовательности.

5.2 Реализация на Python

```python
import numpy as np
import matplotlib.pyplot as plt

# Parameters for the Mandelbrot set
width, height = 800, 600
max_iter = 100
xmin, xmax = -2.5, 1
ymin, ymax = -1, 1

# Create the complex plane
x = np.linspace(xmin, xmax, width)
y = np.linspace(ymin, ymax, height)
X, Y = np.meshgrid(x, y)
C = X + 1j * Y

# Initialize arrays
Z = np.zeros_like(C)
M = np.full(C.shape, True, dtype=bool)
iterations = np.zeros(C.shape, dtype=int)

# Iterative process
for i in range(max_iter):
       Z[M] = Z[M] ** 2 + C[M]
       escaped = np.abs(Z) > 2
       iterations[M & escaped] = i
       M[M & escaped] = False

# Visualization
plt.figure(dpi=100)
plt.imshow(iterations, extent=(xmin, xmax, ymin, ymax), cmap='hot')
plt.xlabel('Re')
plt.ylabel('Im')
plt.title('Mandelbrot Set')
plt.show()
```

5.3 Результаты визуализации

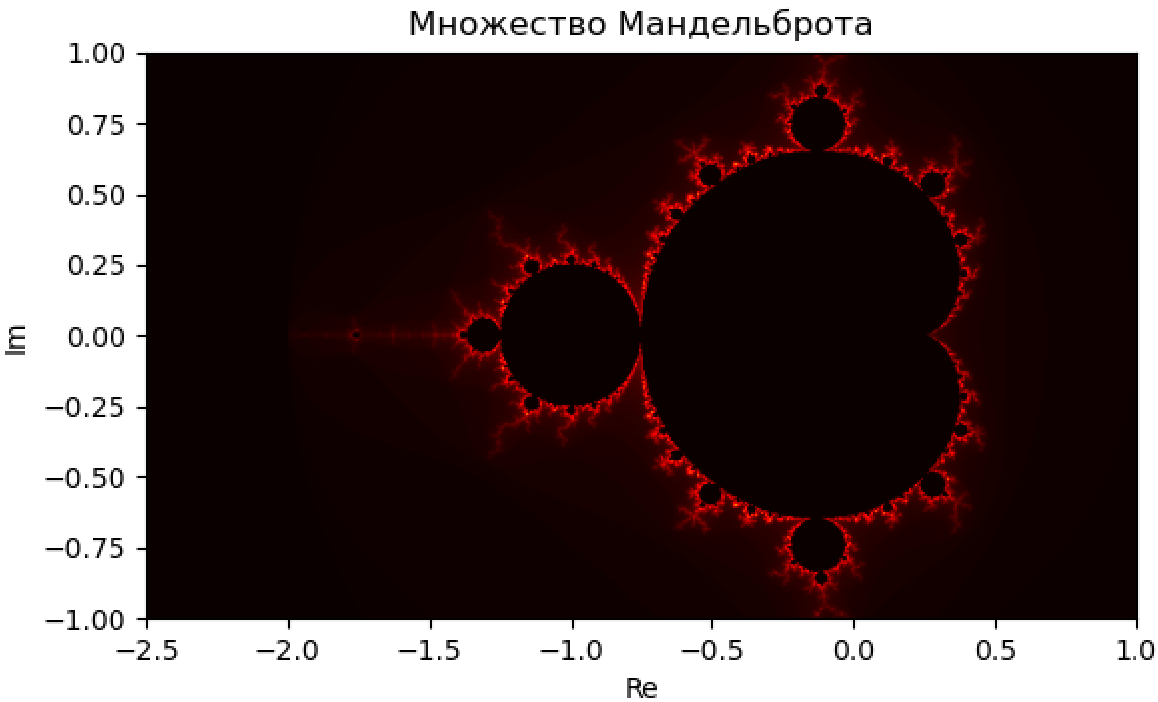

Рис. 1: Множество Мандельброта при максимальном числе итераций 100

6 Визуализация множества Жюлиа

6.1 Описание алгоритма

Алгоритм для визуализации заполненного множества Жюлиа аналогичен алгоритму для множества Мандельброта, с той разницей, что параметр c фиксирован, а начальные значения z_0 берутся из комплексной плоскости.

1. **Определение области комплексной плоскости:** Обычно выбирается квадратная область, например, $x, y \in [-1.5, 1.5]$.

2. **Итерационный процесс:** Для каждого $z_0 = x + iy$ выполняем итерации:

$$z_{n+1} = z_n^2 + c, \tag{10}$$

   где c — фиксированный параметр.

3. **Определение цвета пикселя:** Аналогично множеству Мандельброта, определяем принадлежность точки множеству Жюлиа и окрашиваем пиксель.

6.2 Реализация на Python

```python
import numpy as np
import matplotlib.pyplot as plt

# Parameters for the Julia set
width, height = 800, 800
max_iter = 200
xmin, xmax = -1.5, 1.5
ymin, ymax = -1.5, 1.5
c = -0.5251993 + 0.5251993j # Fixed parameter c

# Create the complex plane
x = np.linspace(xmin, xmax, width)
y = np.linspace(ymin, ymax, height)
X, Y = np.meshgrid(x, y)
Z = X + 1j * Y

# Initialize arrays
M = np.full(Z.shape, True, dtype=bool)
iterations = np.zeros(Z.shape, dtype=int)

# Iterative process
for i in range(max_iter):
       Z[M] = Z[M] ** 2 + c
       escaped = np.abs(Z) > 2
       iterations[M & escaped] = i
       M[M & escaped] = False

# Visualization
plt.figure(dpi=100)
plt.imshow(iterations, extent=(xmin, xmax, ymin, ymax), cmap='twilight_shifted')
plt.xlabel('Re')
plt.ylabel('Im')
plt.title(f'Filled Julia Set for c = {c}')
plt.show()
```

6.3 Результаты визуализации

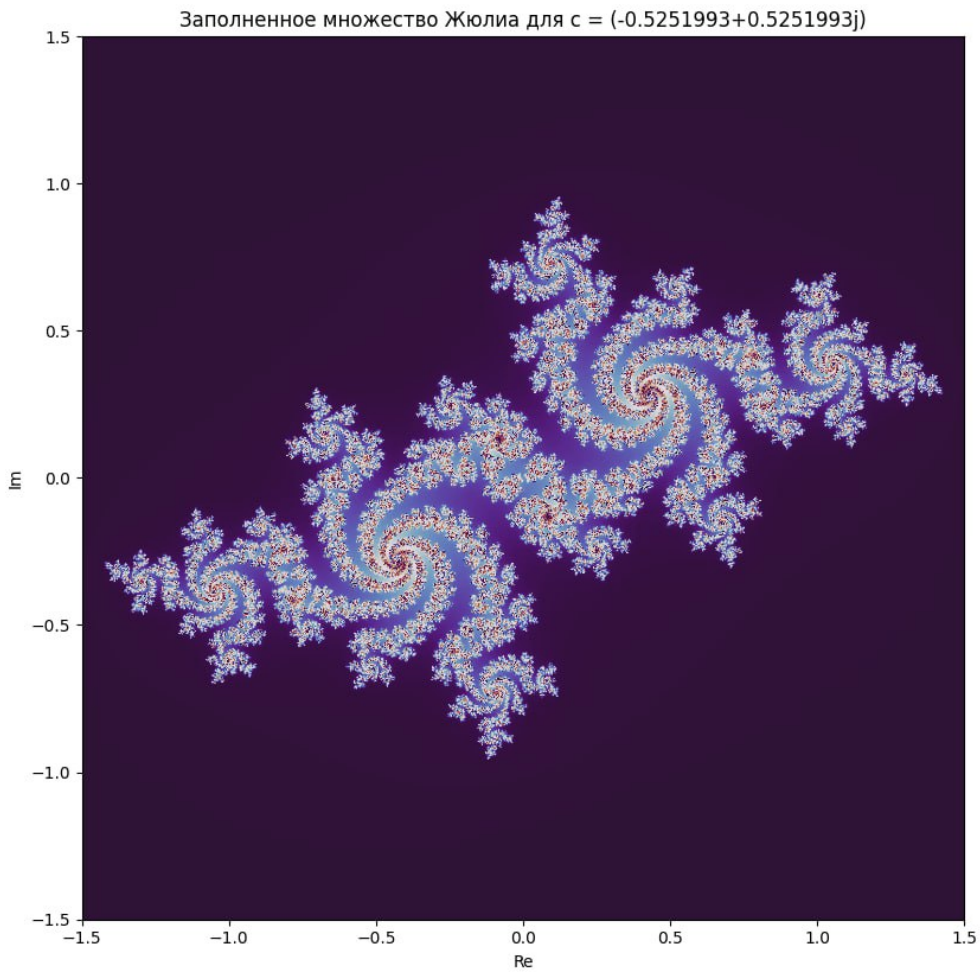

Рис. 2: Заполненное множество Жюлиа при $c = -0.5251993 + 0.5251993i$

7 Исследование другого фрактала: Фрактал Виксека

7.1 Описание фрактала

Фрактал Виксека отличается своей простотой в построении и удивительной самоподобной структурой. Он создаётся путем рекурсивного деления квадрата на пять равных частей и удаления центральной части на каждом шаге итерации. В результате образуется структура, которая напоминает крест или плюс, обладающая интересными фрактальными свойствами.

7.2 Алгоритм построения

Основная идея построения фрактала Виксека заключается в рекурсивном делении квадрата на более мелкие части, что позволяет создать самоподобную и симметричную структуру.

Правила построения:

- **Начало:** Начинаем с квадрата.
- **Итерация:** Делим текущий квадрат на 3 равных по длине отрезка по горизонтали и вертикали, образуя сетку из 9 меньших квадратов.
- **Удаление:** Удаляем средний квадрат, оставляя четыре угловых квадрата и центральный.
- **Повторение:** Повторяем тот же процесс для оставшихся квадратов на каждом уровне рекурсии.

7.3 Реализация на Python

Ниже представлен пример кода на Python, который визуализирует фрактал Виксека с использованием библиотеки matplotlib и рекурсивной функции для построения фрактала.

```
import matplotlib.pyplot as plt
import matplotlib.patches as patches

def draw_vicsek(ax, x, y, size, level):
        if level == 0:
        # Draw the square
        square = patches.Rectangle((x, y), size, size, linewidth=1, edgecolor='black',  facecolor='black')
        ax.add_patch(square)
        else:
        new_size = size / 3
        # Coordinates for the central and corner squares
        positions = [
        (x, y), # bottom left
        (x + 2 * new_size, y), # bottom right
        (x, y + 2 * new_size), # top left
        (x + 2 * new_size, y + 2 * new_size), # top right
        (x + new_size, y + new_size) # center
        ]
        for (nx, ny) in positions:
        draw_vicsek(ax, nx, ny, new_size, level - 1)

def plot_vicsek(level):
        fig, ax = plt.subplots()
        ax.set_aspect('equal')
        ax.axis('off')

        # Initial coordinates and size
        x, y = 0, 0
        size = 1

        draw_vicsek(ax, x, y, size, level)

        plt.show()

if __name__ == "__main__":
```

```
# Recursion level (the higher, the more detailed)
recursion_level = 4
plot_vicsek(recursion_level)
```

7.4 Результаты визуализации

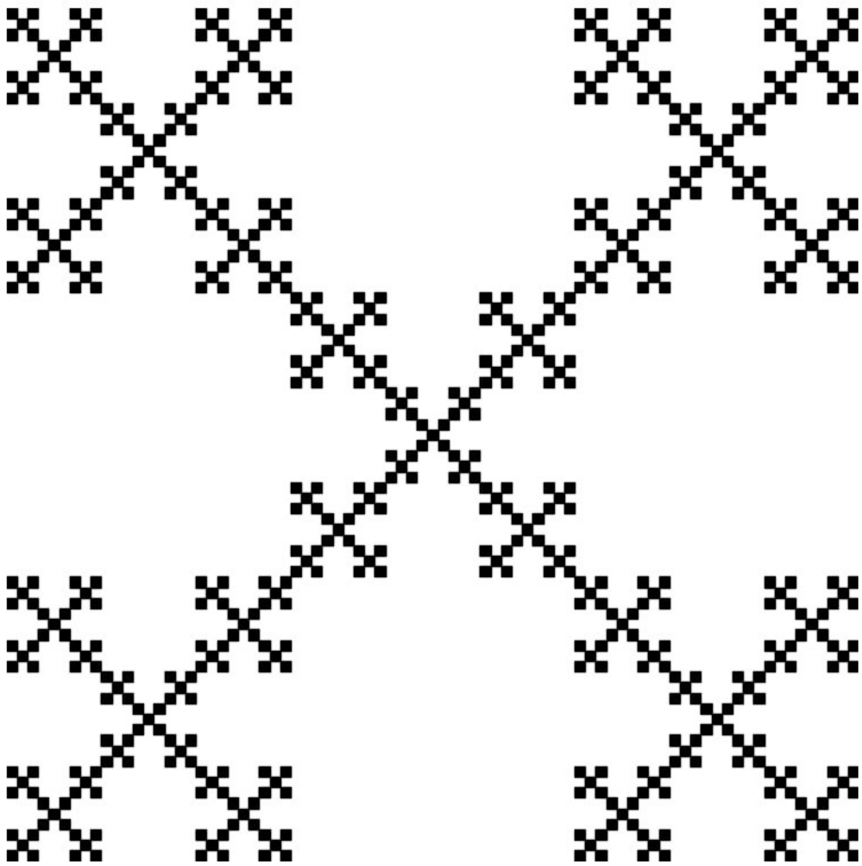

Рис. 3: Фрактал Виксека на уровне рекурсии 4

lab1

7.5 Анализ результатов

Самоподобная структура: Фрактал Виксека демонстрирует самоподобную структуру на каждом уровне рекурсии. С увеличением уровня детализации фрактал приобретает все более сложные и красивые геометрические узоры.

Симметрия и простота: Основной особенностью фрактала Виксека является его симметрия и относительная простота построения. Даже при высоком уровне рекурсии структура остаётся легко узнаваемой, создавая уникальный геометрический рисунок, который можно наблюдать в различных масштабах.

8 Набор изображений при разных параметрах

8.1 Множество Мандельброта при разных итерациях

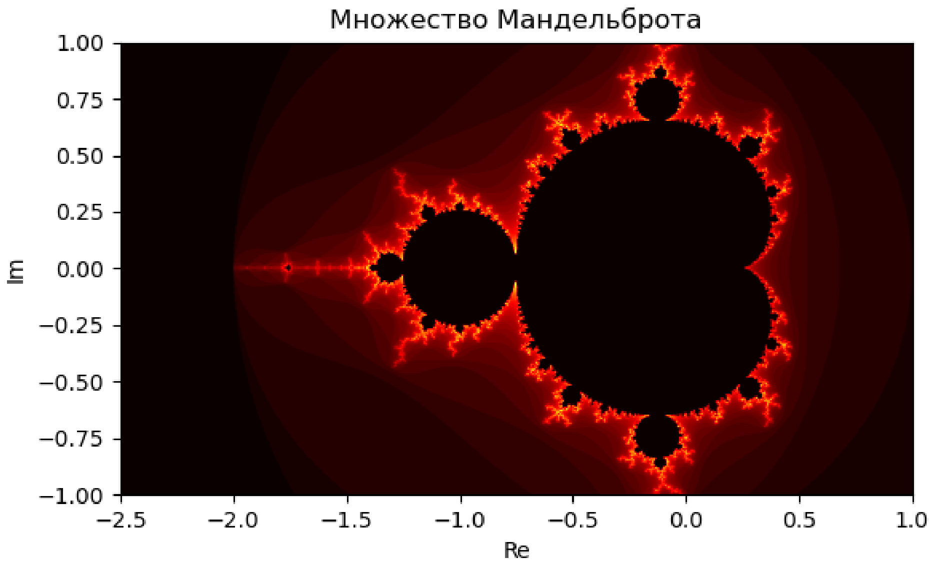

Рис. 4: Множество Мандельброта при max_iter = 50

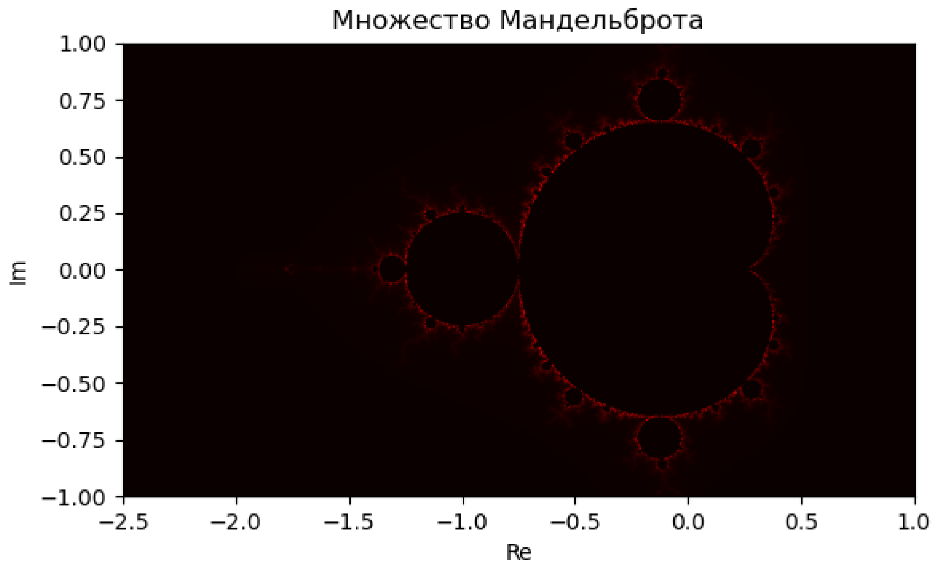

Рис. 5: Множество Мандельброта при max_iter $= 1000$

8.2 Приближение отдельных частей множества Мандельброта

Рис. 6: Приближение области множества Мандельброта

8.3 Множество Жюлиа при разных значениях c

Рис. 7: Множество Жюлиа при $c = 0.285 + 0.01i$

Рис. 8: Множество Жюлиа при $c = -0.70176 - 0.3842i$

9 Заключение

В данной лабораторной работе мы подробно изучили множества Мандельброта и Жюлиа, доказали их основные свойства и реализовали алгоритмы для их визуализации. Мы также исследовали бассейны Ньютона, продемонстрировав фрактальную природу границ областей сходимости метода Ньютона к различным корням.

Проведённые эксперименты показали, как простые итерационные процессы могут приводить к возникновению сложных и красивых фрактальных структур. Изменение параметров итераций и приближения позволило нам наблюдать разнообразие форм и узоров, присущих этим множествам.