

Санкт-Петербургский государственный университет

Прикладная математика и информатика


Отчет по учебной практике 1 (научно-исследовательской работе)

(семестр1)

Построение фрактальных мужеств на языке Python

Выполнил:

Милюшков Георгий Геннадьевич



Научный руководитель:

к.ф.-м.н., PhD, с.н.с.

Мокаев Руслан Назирович

Кафедра прикладной кибернетики

Работа выполнена на достойном уровне

и может быть зачтена с оценкой А



Санкт-Петербург

2020

Введение

Фракталы — это очень интересные и красивые фигуры, которые можно получить используя операции масштабирования и копирования. У этих фигур есть очень интересная особенность, что они имеют свойство самоподобия при приближении. С развитием вычислительной техники, фракталы очень часто рисуются при помощи компьютеров. В этом отчете я представлю мой прогресс по изучению фракталов с помощью языка Python.

Передо мной была поставлена задача построить известные фрактальные множества на языке Python. Предполагается нарисовать шесть фракталов:

- 1) Дракон Хартера — Хетуэя
- 2) Снежинка Коха
- 3) Губка Менгера
- 4) Ковёр Серпинского
- 5) Множество Жюлиа
- 6) Множество Мандельброта

Глава 1. Фрактальные множества

Фрактал — это множество, которое имеет свойство самоподобия. То есть, если взять какую-то фигуру или множество, то оно в основных чертах не изменится или будет совпадать с частями изначальной картинке даже при приближении в любую часть фрактала. Это очень интересная особенность, что даже при приближении сама картинка почти не меняется. И эта особенность не только в математических, геометрических фракталов про которые в основном пойдет речь, но это можно встретить в природе, в разных технологиях, компьютерной архитектуры и в много других сферах.

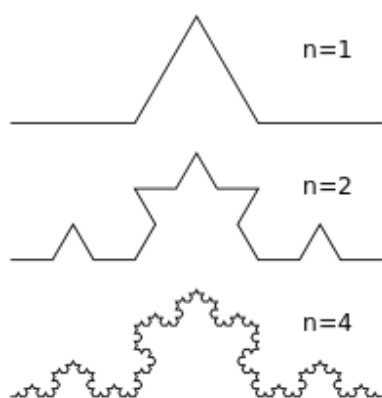


Рисунок 1-Построение снежинки Коха

Очень часто, когда говорят слово «фрактал» подразумеваются геометрические фракталы. Один из самых известных геометрических фракталов, является снежинка Коха, представленная шведским математиком Хельге Фон Кох. Ее принцип очень легко прослеживается на Рисунке 1. К каждой прямой линии дополнительно пририсовывается горка как указано в $n = 1$. Такую горку можно продолжать пририсовывать бесконечно на прямых участках фигуры. При масштабирование данной фигуры, будут заметны исходные горочки и прямые линии. Снежинка Коха, ковер Серпинского и другие геометрические фигуры часто программируются с использованием рекурсии. Функция является рекурсивной если она обращается сама к себе. Часто пользователя просят задать глубину рекурсии. Это означает сколько раз рекурсия будет вызывать саму себя в одной части. Позже это будет мною представлено на примере треугольника Серпинского, где глубина решает сколько треугольников рисовать с одной стороны.

Другим очень популярным фракталом является- множество Мандельброта. Возможно привести пример, допустим есть какое-то число $z = 8$. Если это возвести в квадрат, то получится 64, дальше если возвести 64 в квадрат, получится 4096, если возвести 4096 в квадрат, то получится 16 777 216. Так можно продолжать бесконечно, и в итоге получится бесконечное большое число, даже если начальное число было отрицательным. Если взять -8, то далее все равно будет 64, 4096 и так далее. То есть эта последовательность не ограничена. Но можно начать с числа, с которого последовательность чисел при возведении в квадрат будет ограничена. В расчёт не берется единица, так как при возведении ее в степень n раз, все равно будет 1. Если взять число с модулем меньше единицы, то при возведении его в степень оно будет стремиться к нулю по модулю, то есть оно будет как раз ограничено. Например, рассмотрим число 0.5. Если это число возвести в квадрат, то получится 0.25. Далее при

возведении 0.25 в квадрат получится 0.0625 потом получится 0.00390 и т.д. Последовательность стремится к нулю и ограничена. В данном примере мы считали по формуле $z_{n+1} = z_n^2$. Это также возможно представить в комплексных числах.

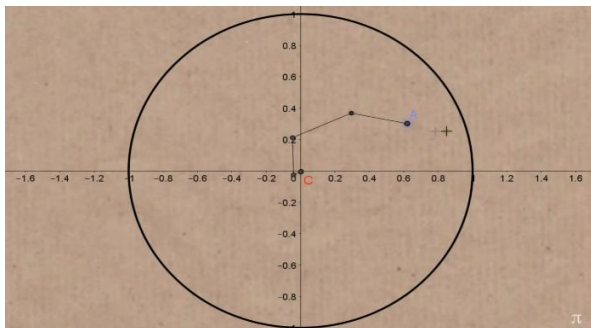


Рисунок 2, границы комплексных чисел при которых последовательность ограничена, и пример точ.А, которая стремится к 0.

При использовании комплексных чисел получится круг с модулем 1 как указано на Рисунке 2. Также на этом рисунке продемонстрирована пример с произвольной точкой А в пределах окружности, которая стремится к 0 при возведении в степень. Точки на рисунке 2 символизируют следующие значения при возведении в квадрат. Если же взять начальную точку за пределами этой окружности, то последовательность будет не ограничена.

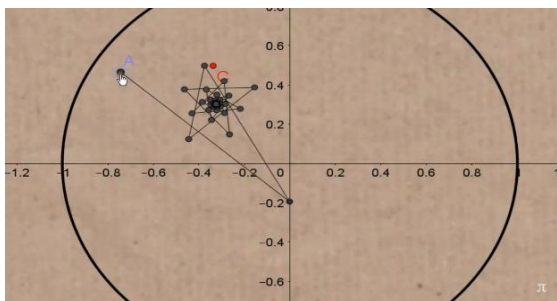


Рисунок 3, последовательность точек на комплексной плоскости в пределах, посчитанная по формуле $z_{n+1} = z_n^2 + t$

Данные примеры были посчитаны по формуле $z_{n+1} = z_n^2$ или по формуле $z_{n+1} = z_n^2 + t$, где $t = 0$. Рисунок последовательности точек при возведении в квадрат сильно изменится при использовании t не равное нулю. Если будет прибавляться какое-то комплексное число t , то возможно получить картинку как указано на Рисунке 3. Последовательность по-прежнему ограничена, но стремится к другому числу. При использовании t не равное нулю, становится не очень понятно, когда наша последовательность будет ограничена, а когда нет. В некоторых

местах за кругом последовательность все равно ограничена, а в некоторых нет. Когда $t = 0$, то границы, когда последовательность ограничена, есть круг с радиусом 1, но когда меняется значение t , то и границы меняются в зависимости от значения t . Эти границы для комплексного числа, с которого мы начинаем возводить в квадрат, которые меняются в зависимости от t , называется множество Жюлиа. Бенуа Мандельброт, французский математик исследовал границы, когда последовательность будет ограниченной если всегда будут, начинать возводить в квадрат с числа ноль, и только менять значения t . То есть он исследовал какие границы для t , если последовательность начинается с 0. Например: допустим $t = -0,5$ - вещественное число то получится: $0^2 - 0,5 = -0,5$. Далее: $0,5^2 - 0,5 = -0,25$, далее: $-0,25^2 - 0,5 = -0,4375$, далее: $-0,4375^2 - 0,5 = -0,30879$. Этот пример показывает, что последовательность чисел, высчитанная по формуле $z_{n+1} = z_n^2 + t$, при константе $t = -0,5$ не растет бесконечно и является ограниченной. Для наглядности, если взять положительное 0,5 то оно будет расти бесконечно: $0^2 + 0,5 = 0,5$, $0,5^2 + 0,5 = 0,75$, $0,75^2 + 0,5 = 1,0625$, $1,0625^2 + 0,5 = 1,6289$, $1,6289^2 + 0,5 = 3,15$. Так можно продолжать бесконечно, и получится бесконечное большое число. Это уже не круг как было раньше, а некоторая другая форма. Оказывается, это фрактал, названный в честь, самого открывателя- множество Мандельброта. Все что было выше представлено, можно написать в одном предложении: множество Мандельброта — это множество таких точек t на комплексной плоскости, для которых рекуррентное соотношение $z_{n+1} = z_n^2 + t$, при $z_0 = 0$ задает ограниченную последовательность.

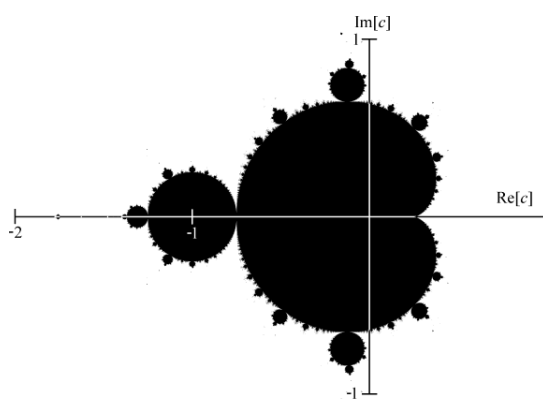


Рисунок 4, множество Мандельброта

Это множество изображено на Рисунке 4. Но это не просто множество, а фрактал, так как при приближении в одну из этих маленьких окружностей на картинке, будет видно частичное подобие изначальной картинки.

Как мною было упомянуто в самом начале, фракталы бывают очень разными и окружают нас повсюду. Например снежинка, листья или география расположения берегов и гор часто напоминают фракталы. Даже молния и дыхательная система у людей и животных есть много схожего с фракталами. В этом отчете было подробно рассказано о двух фракталах, тип которых составляют ключевую часть данной работы.

Глава 2. Программирование

Подглава 1. Python

На данный момент мною изучены основы языка Python. Были выполнены следующие работы:

- Изучено в какой среде писать, Spyder, Pycharm или Thonny. Мною выбрано было Spyder.
- Изучен синтаксис и основные математические операторы.
- Изучено как задавать переменные и ими пользоваться.
- Изучено пользование разных тип данных. Было написано, разные, программы используя тип данных str, int, float, bool, np.array.
- Изучено, как получить данные от пользователя используя оператор input.
- Изучено, как проводить проверки true и false используя операторы if, else и ==, !=, <=, <, >=, >
- Изучены и написаны программы используя циклы for и while.
- Изучено работы с функциями. Передачу и получение данных из функций.
- Изучено пользования массивов и команд .append, .extend, .sort, .reverse.
- Написаны простейшие программы используя библиотеку numpy. Были в основном использованы команды np. arange, np.linspace, np.zeros.
- Написаны простейшие программы используя библиотеку matplotlib. Также было изучено использование соответствующих команд для стилистики графиков.

Подглава 2. Визуализация

Во время изучения Python мною было написано много разных программ в том числе фрактал-треугольник Серпинского. Код этой программы находится в приложении А. Пользователя

спрашивают глубину рекурсии. Рисунок 5, это пример данного фрактала с рекурсией глубиной 7. То есть рисуются 7 треугольников с каждой стороны.

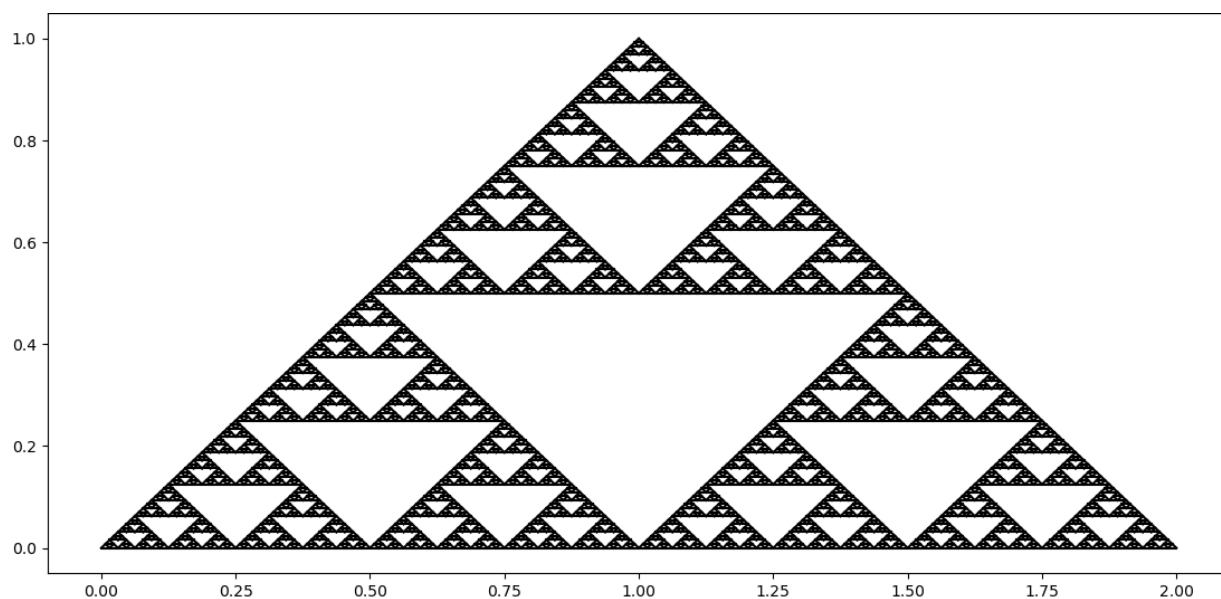


Рисунок 5, треугольник Серпинского, глубина 7

В рисунке 6 продемонстрировано, что будет если приблизить в самый нижний левый треугольник который идет по координатам 0,0-0.5 по оси x .

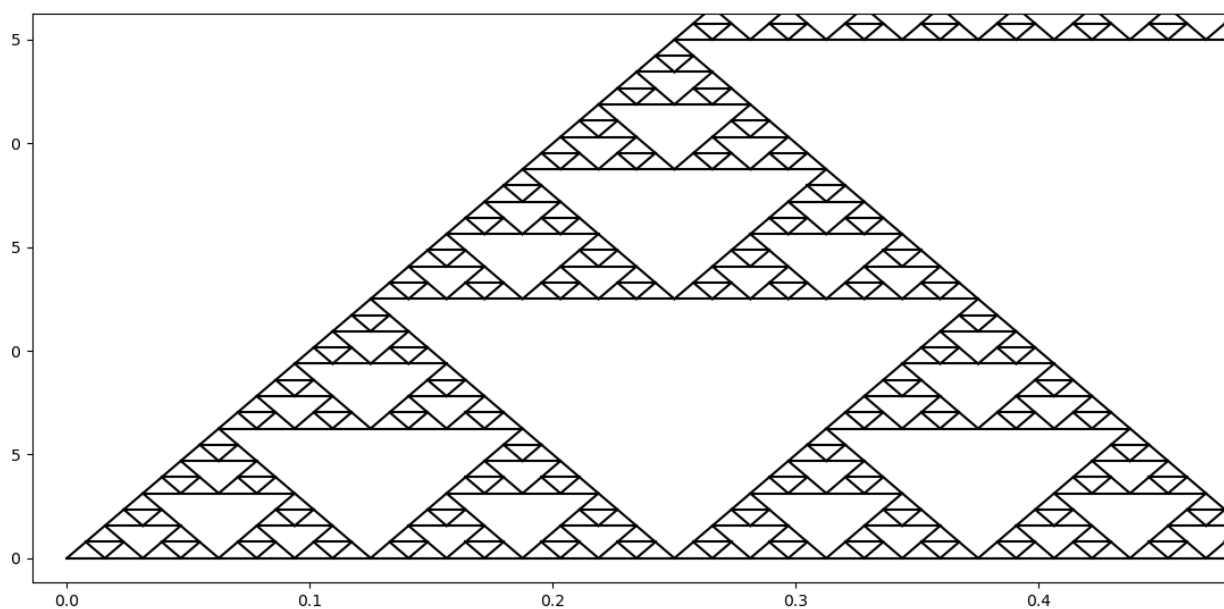


Рисунок 6, треугольник Серпинского, приближение рисунка 5, в координаты по оси x 0,0-0,5

Возможно заметить, что фрактал почти не изменился. Также можно приблизить в координаты 0,0 - 0,15

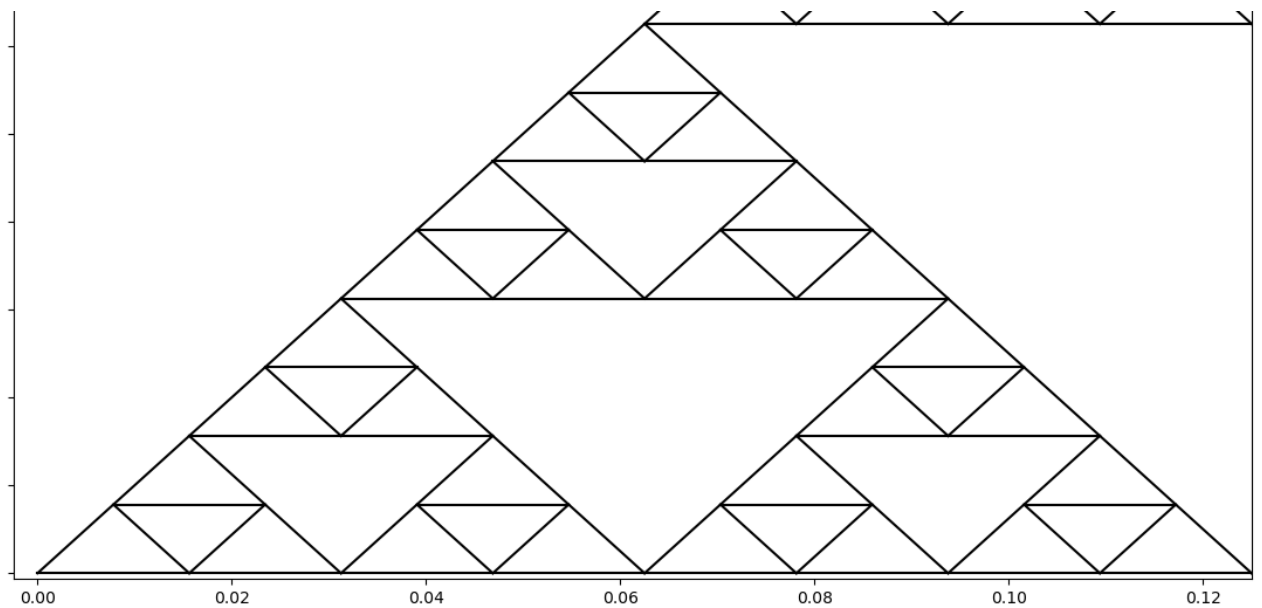


Рисунок 7, треугольник Серпинского, приближение рисунка 6, в координаты по оси x 0,0-0,15

Заключение

В этом семестре мною была выполнена работа по изучению разных фракталов и изучено основы программирования, дающие достаточное количество знаний для построения заданных фрактальных множеств. В отчете было представлено прогресс по изучению языка Python, было подробно рассказано о двух фракталах и представлен промежуточный результат в виде треугольника Серпинского.

В будущем планируется проиллюстрировать все заданные фракталы, написать про фрактальные множества и про математическое моделирование на языке Python. Остается основная часть курсовой работе, но большая часть для достижения этой цели уже выполнена.

Список литературы

1. Skrindo Knut, Innføring i Python: учеб.пособие/ 2019, Kopinor-52 с.
2. Saha Amit, Doing math with Python / 2015, No starch Press, USA, 265 стр. URL: <http://index-of.es/Varios-2/Doing%20Math%20with%20Python.pdf>

Приложение А. Код

```
1 import matplotlib.pyplot as plt
2
3
4 def podder(kol,x1,y1,x2,y2,x3,y3,n):
5     plt.plot([x1,x2],[y1,y2],'k')
6     plt.plot([x1,x3],[y1,y3],'k')
7     plt.plot([x2,x3],[y2,y3],'k')
8
9     if(kol<n):
10         podder(
11             kol+1,
12             (x1 + x2) / 2 + (x2 - x3) / 2,
13             (y1 + y2) / 2 + (y2 - y3) / 2,
14             (x1 + x2) / 2 + (x1 - x3) / 2,
15             (y1 + y2) / 2 + (y1 - y3) / 2,
16             (x1 + x2) / 2,
17             (y1 + y2) / 2,
18             n)
19
20         podder(
21             kol+1,
22             (x3 + x2) / 2 + (x2 - x1) / 2,
23             (y3 + y2) / 2 + (y2 - y1) / 2,
24             (x3 + x2) / 2 + (x3 - x1) / 2,
25             (y3 + y2) / 2 + (y3 - y1) / 2,
26             (x3 + x2) / 2,
27             (y3 + y2) / 2,
28             n)
29
30         podder(
31             kol+1,
32             (x1 + x3) / 2 + (x3 - x2) / 2,
33             (y1 + y3) / 2 + (y3 - y2) / 2,
34             (x1 + x3) / 2 + (x1 - x2) / 2,
35             (y1 + y3) / 2 + (y1 - y2) / 2,
36             (x1 + x3) / 2,
37             (y1 + y3) / 2,
38             n)
39
40
```

```
41
42 def glavder(x1,y1,x2,y2,x3,y3,n):
43     plt.plot([x1,x2],[y1,y2],'k')
44     plt.plot([x1,x3],[y1,y3],'k')
45     plt.plot([x2,x3],[y2,y3],'k')
46
47     z=1
48     ax=(x1+x2)/2
49     ay=(y1+y2)/2
50     bx=(x1+x3)/2
51     by=(y1+y3)/2
52     cx=(x2+x3)/2
53     cy=(y2+y3)/2
54     return podder(z,ax,ay,bx,by,cx,cy,n)
55
56 n=int(input("Glubina rekursii:"))
57
58
59 glavder(0,0,2,0,1,1,n)
```