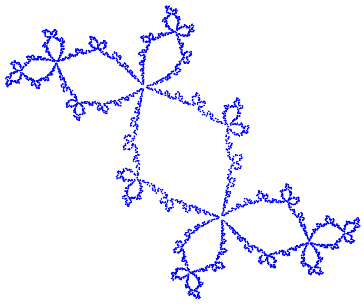
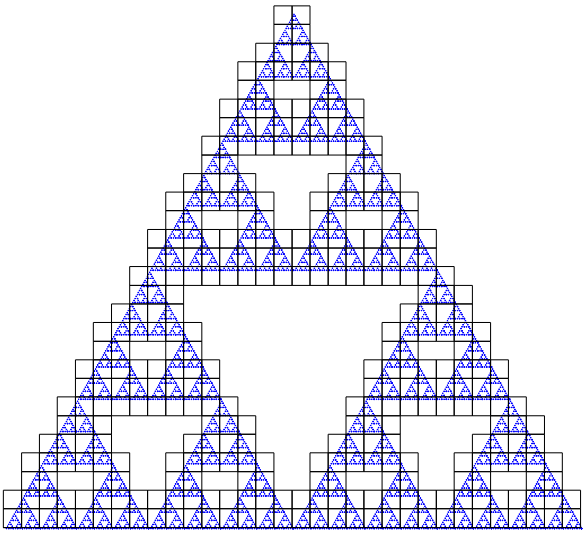
rijksuniversiteit groningen
faculteit Wiskunde en Natuurwetenschappen

Dimension of Fractals





Bacheloronderzoek Wiskunde

November 2014

Student: Joël Grondman

Eerste Begeleider: Alef E. Sterk

Tweede Begeleider: Arjan v.d. Schaft

Introduction

In this Bachelor thesis we will look at the chaotic behaviour of rational functions on the complex plane which introduces Julia sets. Julia sets are sets that do not simplify when magnified, moreover the whole set repeats itself on magnification. Sets with this trait are called fractals. Aside from Julia sets there are other fractals such as the Cantor set and the Pascal triangle which also do not simplify under magnification.
A fractal is often described by its dimension. Topological dimension preserves dimension under two homeomorphic spaces however, there are fractals for which topological dimension does not make sense. As an example a Hilbert curve [8] fills the whole plane and is homeomorphic to the unit interval causing a contradiction when topological dimension is applied. This thesis introduces fractal dimension for which there are several theorems which give the exact dimension for a specific group of fractals as well as numerical methods which approximate the fractal dimension. We will be focusing on the latter.

Chapter 1 will be devoted to the basic properties of Julia sets as well as numerical methods that follow from these properties. Chapter 2 introduces several new dimensions we will be using to describe fractals aswell as a few examples on other sets. Chapter 3 will focus on the numerical approximation of dimensions and provide examples and results using these approximations. For our algorithms we will be using Matlab and the code can be found in the appendix.

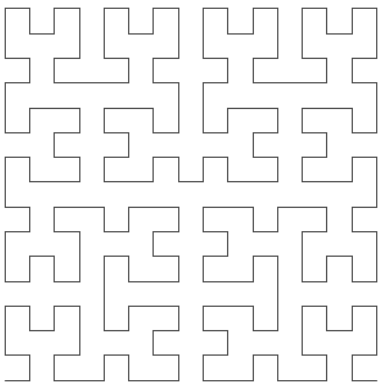


Figure 1: One example of a Hilbert Curve.

Contents

Introduction **2**

1 Julia and Fatou sets **4**
1.1 Definition . . . 4
1.2 IIM & MIIM . . . 6
1.3 BSM & MBSM . . . 7

2 Alternative Dimensions **9**
2.1 Hausdorff Dimension . . . 9
2.2 Boxcounting Dimension . . . 12
2.3 Boxcounting Dimension of self-similar sets . . . 12
2.4 Pointwise Dimension . . . 13
2.5 Generalized Boxcounting Dimension . . . 14

3 Numerical Estimates **15**
3.1 Correlation Dimension . . . 15
3.1.1 General Correlation Dimension . . . 16
3.1.2 q-point Correlation . . . 16
3.2 Boxcounting Algorithm . . . 19
3.3 Results . . . 21
3.3.1 Cantor dust & Sierpinski triangle . . . 21
3.3.2 Julia sets . . . 22

Conclusion **24**

A Matlab Code **25**
A.1 Correlation dimension . . . 25
A.2 Boxcounting dimension . . . 26
A.3 Cantor set & Cantor Dust . . . 27
A.4 Pascal triangle . . . 27
A.5 IIM & MIIM . . . 28
A.6 BSM & MBSM . . . 30

Bibliography **34**

Chapter 1

Julia and Fatou sets

1.1 Definition

The Julia set is a set on the Riemann sphere of which the behaviour is solely described by a function f in the following definition:

Definition 1.1. (Julia and Fatou sets [2]) Let $f : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$ be a non-constant holomorphic function and $f^n : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$ be its n-fold iterate. For $z_0 \in \hat{\mathbb{C}}$ if there is a neighboorhood U of z_0 such that f^n forms a normal family restricted on U then z_0 is a regular point and belongs to the Fatou set of f denoted by F_f. Otherwise z_0 belongs to the Julia set J_f.

We call $z_0, f(z_0), f^2(z_0), ..., f^n(z_0)$ an orbit. Roughly speaking $\{f^n\}$ is a normal family restricted on U when all orbits $f^n(z_0)$ $z_0 \in U$ converges uniformly to another bounded orbit or to ∞. This definition is not easy to grasp, but if f is a polynomal we can use an equivalent definition:

Definition 1.2. (Filled Julia sets [2]) Let $f : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$ be a polynomal function. The filled Julia set K_f is defined as $K_f = \{z \in \hat{\mathbb{C}} : f^n(z) \not\to \infty \text{ as } n \to \infty\}$. The Julia set J_f is defined as the boundary of the filled Julia set: $J_f = \partial K_f$.

Equivalently the Julia set J_f can be described as the boundary between points that are bounded and points that go to ∞ under iteration of f.

As an example consider the function $f(z) = z^2$. Any point inside the unit circle will remain in the unit circle and the points outside will go to ∞ hence the filled Julia set K_f is the unit disc. It follows that the Julia set J_f is the unit circle. The first definition of Julia sets will result to the same conclusion as any family $\{f^n\}$ is normal restricted on U where U does not intersect with the unit circle, when U does intersect with the unit circle we can find two orbits that converge to ∞ and to 0 respectively and conclude that the family $\{f^n\}$ is not normal.

Theorem 1.3. *Suppose $z_0 \in J_f$ then the set $\{z \in \hat{\mathbb{C}} | f^n(z) = z_0$ for some $n \geq 0\}$ is dense in J_f.*

Given an stable fixed point z_0 ($z_0 = f(z_0)$ and $|f'(z_0)| < 1$) the basin of attraction of z_0 is defined as $A(z_0) = \{z \in \hat{\mathbb{C}} | f^n(z) \to z_0 \text{ as } n \to \infty\}$.

Theorem 1.4. *If $f : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$ is a rational function and z_0 an stable fixed point then $\partial A(z_0) = J_f$.*

Given an orbit $z, f(z), f^2(z), ..., f^{n-1}(z), f^n(z) = z$ of period n, this orbit is attracting if $|(f^n)'(z)| < 1$ which implies that f^n contains n stable fixed points. Our definition of Julia sets implies $J_f = J_{f^n}$ meaning that if z_0 is a stable fixed point of f^n then $\partial A(z_0) = J_f$ aswell. For proofs of the above theorems and more information about Julia sets we refer to [2].

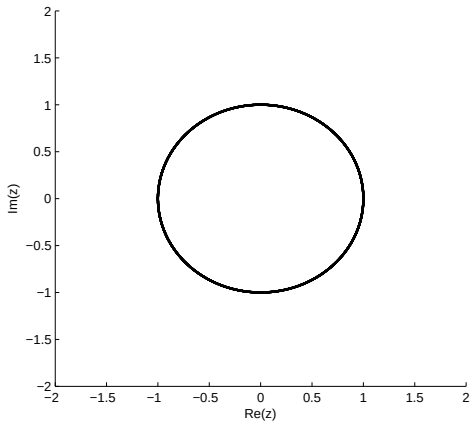


Figure 1.1: The Julia set for $f(z) = z^2$.

1.2 IIM & MIIM

Given that the set of pre-images of a point in J_f is dense in J_f we can generate a Julia set for given function f and $z_0 \in J_f$ by finding the pre-image. This method is called IIM, the inverse iteration method.

This method starts with a single point in the Julia set and computes the pre-image to create new points, the pre-image is applied to every new point several times creating the set. For example: for the function $f(z) = z^2 + c$ the pre-image $f^{-1}(z) = \sqrt{z - c}$ and $f^{-1}(z) = -\sqrt{x - c}$ are both applied to every new point. The downside to this method is that points tend to iterate to limits within the Julia set and causes the plot to be dense in certain areas of the actual Julia set while some areas do not contain any points at all as the iteration needs to be cut off at some point due to the exponentially increasing amount of pre-images.

A solution to this problem is MIIM, the modified inverse iteration method. For our modification we apply a grid of boxes of a certain size on the Julia set. We apply a maximum limit of points in each box. IIM is still applied repeatively but if a new point would be added to a box which has reached its limit, the point is discarded and thus its pre-images will not be calculated. This method ensures that the density of points in every part of a Julia set remains relatively the same.
There are other modifications on the IIM such as selecting a random pre-image instead of using both or discarding a random selection of points from a box when it has reached its limit instead.
The downside to MIIM is that not every part of the Julia set is evenly represented as can be seen from the IIM method in figure 1.3 and this carries over to MIIM.

1.3 BSM & MBSM

Given the basin of attraction of an attractive fixed point in J_f then the boundary of the basin equals J_f. Utilizing this fact we can determine whether a point z is in J_f by looking at how close z is to different basins of attraction. This method is called BSM, the boundary scanning method.

Given a grid of $N \times N$ pixels covering the Julia set we take the vertices of each pixel and label the vertices according to the basin they are attracted to under iteration of f. If the vertices are labeled differently then the corresponding pixel must be close to two different boundaries and we assign this pixel to the Julia set. After repeating this process for all the pixels we obtain a list of pixels that approximates the Julia set. This approximation depends on the maximum number of iterations we allow and the size of the pixel. The computing time of this algorithm is too high as given a grid of N^2 pixels we have to iterate $(N+1)^2$ vertices on this grid.

Instead of a grid we start with one pixel contained in the Julia set and determine whether the neighbouring pixels will be assigned to the Julia set or not by the same method as before. This is repeated for all the newly assigned pixels to the Julia set. This method is only possible for connected Julia sets which can be easily verified for polynomials as the Julia set is connected when the finite critical points of f are bounded under iteration of f [2]. There are exceptions to this such as the Julia set corresponding to $f(z) = z^2 + i$ which is connected but only has one attractive basin making the BSM ineffective.
Depending on the Julia set the MBSM may not represent the Julia set completely or not at all. This can be caused by the low amount of iterations, the seed or how the neighbouring pixels are defined. The solution to this is to change these parameters.

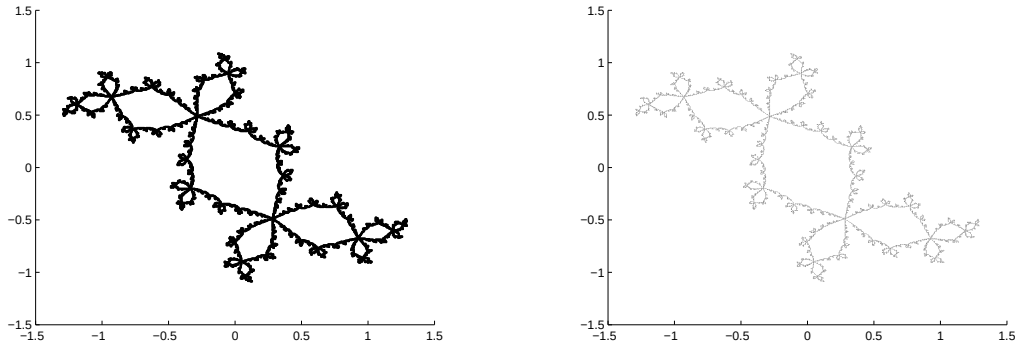


Figure 1.2: The Julia set for $f(z) = z^2 - 0.123 + 0.745i$ also known as the Douady rabbit generated using BSM (left) and MBSM (right). Both algorithms lead to approximately the same result but the MBSM is significantly faster.

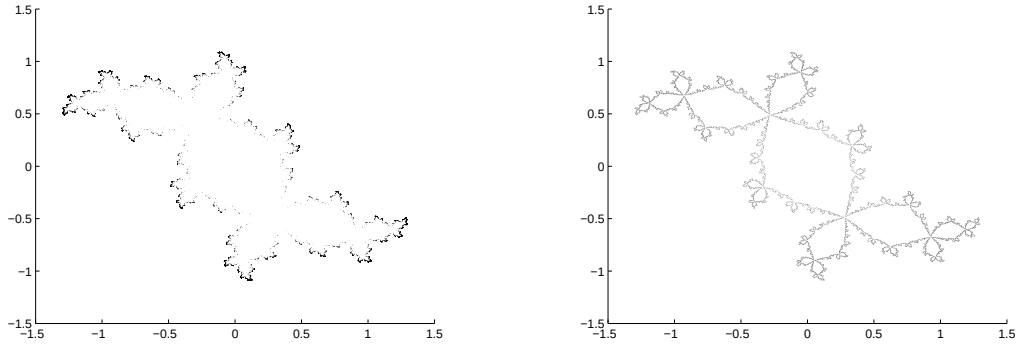


Figure 1.3: The left side is using IIM with 16383 points and MIIM on the right side using a grid of boxes of size 0.005 with a limit of one point in each box leading to an approximation of 8848 points.

Chapter 2

Alternative Dimensions

The dimension of a set can be defined in different ways. One example of a dimension is the topological dimension. As the name suggests the topological dimension of a set is a topological property, if f is a homeomorphism then the topological dimension of A and $f(A)$ are equal. However, this property of topological dimension comes with consequences. The Moore curve has topological dimension 1 but it fills the entire plane which has topological dimension 2. Therefore we will use a different definition of dimension, the Hausdorff dimension.

2.1 Hausdorff Dimension

Definition 2.1. (Hausdorff Measure [4]) We define the diameter of a non-empty subset $U \subset \mathbb{R}^n$ as $|U| = \sup\{|x-y| : x, y \in U\}$ where $|x-y| = (\sum_{i=1}^{n} |x_i - y_i|)^{1/2}$ is the *Euclidean distance*. We say that the collection of sets $\{U_i\}$ is a δ-cover of F if $F \subset \cup_{i=1}^{\infty} U_i$ and $|U_i| \leq \delta$ for all i. Suppose F is a subset of $\mathbb{R}^n$ and s is a nonnegative number. For any $\delta > 0$ we define:

$$H^s_\delta(F) = \inf\{\sum_{i=1}^{\infty} |U_i|^s : \{U_i\} \text{ is a } \delta\text{-cover of } F\}.$$

We define the s-dim Hausdorff measure of F as $H^s(F) = \lim_{\delta \to 0} H^s_\delta(F)$.

Theorem 2.2. *[4] The s-dimensional Hausdorff measure is a measure on $\mathbb{R}^n$.*

Proof. A function m is a meaure on $\mathbb{R}^n$ that assigns a nonnegative value to sets with the following properties:

- $m(\emptyset) = 0$;
- $m(A) \leq m(B)$ if $A \subset B$;

- $if A_1, A_2, ...$ is a countable sequence of sets then

$$m(\bigcup_{i=1}^{\infty} A_i) \leq \sum_{i=1}^{\infty} m(A_i)$$

  with equality when A_i are disjoint Borel sets.

A Borel set is defined as follows:

- Every open and closed set is a Borel set;
- The union/intersection of a countable collection of Borel sets is a Borel set.

H^s satisfies the first two conditions. The equality condition for disjoint Borel sets is trickier. Let $A = \cup_{i=1}^{\infty} A_i$ be a countable union of disjoint Borel sets. Then:

$$H^s_\delta(A) = \inf\{\sum_{i=1}^{\infty} |U_i|^s : \{U_i\} \text{ is a } \delta\text{-cover of } A\}.$$

As $\delta \to 0$ we can decompose the cover $\{U_i\}$ into specific covers for each A_i. $\{U_i\} = \{U_i\}_{A_1} \cup \{U_i\}_{A_2} \cup ...$ where $\{U_i\}_{A_j}$ is a δ-cover for A_j.

$$H^s_\delta(A) = \inf\{\sum_{i=1}^{\infty} |U_i^{A_1}|^s + \sum_{i=1}^{\infty} |U_i^{A_2}|^s + ... : \{U_i^{A_j}\} \text{ is a } \delta\text{-cover of } A_j\}$$

The different covers have no influence on each other as $\delta \to 0$ thus:

$$H^s(A) = H^s(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} (A_i).$$

□

For different values of s different measures can be found. As a function of s the s-dimensional Hausdorff measure decreases when s increases. Now if $t > s$ and $\{U_i\}$ a δ-cover of F we have:

$$\sum_i |U_i|^t = \sum_i |U_i|^{t-s} |U_i|^s \leq \delta^{t-s} \sum_i |U_i|^s.$$

Taking the infimum over all possible covers we get $H^t_\delta(F) \leq \delta^{t-s} H^s_\delta(F)$. Letting $\delta \to 0$, if $H^s(F) < \infty$ then $H^t(F) = 0$. The *critical point* s at which $H^t(F) = \infty$ for $t < s$ and $H^t(F) = 0$ for $t > s$ is called the Hausdorff dimension.

Definition 2.3. (Hausdorff Dimension [4]) The Hausdorff dimension of F is defined as:

$$\dim_H(F) = \inf\{s \geq 0 | H^s(F) = 0\} = \sup\{s \geq 0 | H^s(F) = \infty\}.$$

Example 2.4. Let F be the rationals of the interval $[0,1]$. We will show that for all $s > 0$ $H^s(F) = 0$ which implies $\dim_H(F) = 0$.
Since F is countable we can describe the set as $F = \{f_1, f_2, ...\}$. For all $\delta > 0$ we define for all $i \geq 0$ U_i as an interval containing f_i with length $(\delta/2^i)^{1/s}$. $\sum_{i=1}^{\infty} |U_i|^s = \sum_{i=1}^{\infty} \delta/2^i = \delta$. Thus $H^s_\delta \leq \delta$ and as $\delta \to 0$ $H^s \leq 0$ for all $s > 0$ which implies $\dim_H(F) = 0$.
The proof of this example can be extended to any countable set which means that any countable set has Hausdorff dimension 0.

Example 2.5. As a consequence of the Hausdorff dimension of the rationals in $[0,1]$ to be zero, the set of irrationals in $[0,1]$ has Hausdorff dimension 1. To see this we will separate the interval $[0,1] = \mathbb{I}[0,1] \cup \mathbb{Q}[0,1]$ where $\mathbb{I}$ and $\mathbb{Q}$ denote the irrational and rational part of the interval respectively. Since H^s is a measure $H^s([0,1]) = H^s(\mathbb{I}[0,1]) \cup H^s(\mathbb{Q}[0,1]) = H^s(\mathbb{I}[0,1])$ and since $\mathbb{I}[0,1] \subset [0,1]$ $H^s(\mathbb{I}[0,1]) \leq H^s([0,1])$ therefore $H^s(\mathbb{I}[0,1]) = H^s([0,1])$ and $\dim_H([0,1]) = \dim_H(\mathbb{I}[0,1])$=1. We have not proven $\dim_H([0,1]) = 1$ yet but it can be easily shown that $H^1([0,1]) = 1$ and since this 1-dimensional Hausdorff measure takes on a value between ∞ and 0 it follows that $\dim_H([0,1]) = 1$.

Theorem 2.6. *[4] (scaling property) Let S be a similarity transformation of scale factor $\lambda > 0$. if $F \subset \mathbb{R}^n$ then*

$$H^s(S(F)) = \lambda^s H^s(F).$$

A similarity transformation is a function that preserves shapes, examples include translation, rotation, reflection and scaling. The function $f : \mathbb{R} \to \mathbb{R}$, $f(D) = 1/4x + 1$ is a similarity transformation that scales the shape D by a factor of $1/4$ and translates it to the right on the real line by 1.

Proof. If $\{U_i\}$ is a δ-cover of F then $\{S(U_i)\}$ is a $\lambda\delta$-cover of $S(F)$, implying

$$\sum_i |S(U_i)|^s = \lambda^s \sum_i |U_i|^s.$$

Taking the infimum over all covers gives $H^s_{\lambda\delta}(S(F)) \leq \lambda^s H^s_\delta(F)$. As $\delta \to 0$ we get $H^s(S(F)) \leq \lambda^s H^s(F)$.
To prove $H^s(S(F)) \geq \lambda^s H^s(F)$ we replace S, F, λ by S^{-1}, $S(F)$, $1/\lambda$ respectively and repeat the above. □

Example 2.7. The Cantor set is made by starting at the interval $[0,1]$ and removing the middle third of it resulting in two new intervals $[0,1/3]$, $[2/3,1]$ this process is then repeated for every new interval resulting in the Cantor set F. The Cantor set can be split in two similar parts $F_L = F \cap [0,1/3]$ and $F_R = F \cap [2/3,1]$ which are scaled down by $1/3$. Thus theorem 2.6 can be applied: $H^s(F) = H^s(F_L \cup F_R) = H^s(F_R) + H^s(F_L) = \frac{1}{3}^s H^s(F) + \frac{1}{3}^s H^s(F)$. Assuming $0 < H^s(F) < \infty$ (which will be proven later on) for $s = \dim_S(F)$ we can divide by H^s to get $1 = 2(\frac{1}{3})^s$ for which the solution is $s = \log 2/\log 3$.

2.2 Boxcounting Dimension

In practice the Hausdorff dimension of a set can be difficult to find and we shall therefore introduce a new dimension: the boxcounting dimension.

Definition 2.8. [4] Let F be a non-empty bounded subset of $\mathbb{R}^n$ and let $N_\delta(F)$ be the smallest number of sets of diameter at most δ which can cover F. Then the box-counting dimension is defined as:

$$\dim_B F = \lim_{\delta \to 0} \frac{\log N_\delta(F)}{-\log \delta}.$$

Intuitively try to cover a set F with sets of diameter at most δ. When the diameter decreases how many sets are then needed to cover F? The exponent s can be seen as the exponential rate at which the amount of sets increases when δ decreases. As δ decreases $N_\delta(F) \sim c\delta^{-s}$ for a certain s, taking logarithms gives $\log N_\delta(F) \sim \log(c) - s\log(\delta)$ and some rearranging gives: $s = \lim_{\delta \to 0} \frac{\log N_\delta(F)}{-\log \delta}$.

Instead of covering a set F by arbitrary sets of diameter at most δ we can define a fixed grid with size δ over the space the set F is in and $N'_\delta(F)$ as the amount of boxes in the grid which contain a point of F, we call the dimension obtained from this grid $\dim_{B'} F$. Obviously $N'_\delta(F) \geq N_\delta(F)$ thus $\dim_{B'} F \geq \dim_B F$. In the other direction each covering set of diameter δ is covered by at most 4 sets in the fixed grid implying $N'_\delta(F) \leq 4N_\delta(F)$.

$$\dim_{B'} F = \lim_{\delta \to 0} \frac{\log N'_\delta(F)}{-\log \delta} \leq \lim_{\delta \to 0} \frac{\log 9N_\delta(F)}{-\log \delta} = \dim_B F.$$

This proves equality $\dim_B F = \dim_{B'} F$. Thus $N_\delta(F)$ can be defined as the number of boxes in a grid with length δ needed to cover F, hence the name boxcounting dimension. This redefinition of the boxcounting dimension lends itself to numerical algorithms.

Example 2.9. As an example the boxcounting dimension for the interval $[0,1]$ is 1 as $N_\delta([0,1]) = \lceil 1/\delta \rceil$ which results in $\dim_B [0,1] = \lim_{\delta \to 0} \frac{\log N_\delta([0,1])}{-\log \delta} = \lim_{\delta \to 0} \frac{-\log \lfloor \delta \rfloor}{-\log \delta} = 1$;

Example 2.10. The rational numbers of the interval $[0,1]$ is dense in $[0,1]$ meaning that we have to cover the entire interval to cover the rational numbers. This implies $N_\delta([0,1]) = \lceil 1/\delta \rceil$ and as in the previous example $\dim_B([0,1] \cap \mathbb{Q}) = 1$.

Earlier on we found the Hausdorff dimension of this set to be zero. The Hausdorff and Boxcounting dimension are therefore two different dimensions. But, as we shall see, there are sets for which these dimensions are equal.

2.3 Boxcounting Dimension of self-similar sets

Some definitions. Let D be a closed subset of $\mathbb{R}^n$. A mapping $S : D \to D$ is a contraction on D if there is a ratio $c \in (0,1)$ such that $|S(x) - S(y)| \leq c|x - y|$ for all $x, y \in D$(S is

a similarity if $|S(x) - S(y)| = c|x - y|$ for all $x, y \in D$). A finite family of contractions $\{S_1, ..., S_m\}$, with $m \geq 2$, is called an iterated function system or IFS. A non-empty compact subset F of D is an attractor if $F = \cup_{i=1}^{m} S_i(F)$

Theorem 2.11. *Let $F \subset \mathbb{R}^n$ be an attractor for IFS $\{S_1, ..., S_m\}$ where for all $i \geq 0$ S_i on $\mathbb{R}^n$ is a similarity. If there exists a non-empty bounded open set $V \subset \mathbb{R}^n$ such that $V \supset \cup_{i=1}^{m} S_i(V)$, then $dim_H(F) = dim_B(F) = s$, where s is the solution of:*

$$\sum_{i=1}^{m} c_i^s = 1.$$

Proof. Proof of this theorem can be found in [4]. □

Example 2.12. For the Cantor set F the IFS is: $S_1 = 1/3x$ and $S_2 = 1/3x + 2/3$ which resembles the Cantor set splitting into two similar parts. For the open set $V = (0, 1)$ $S_1(0, 1) \cup S_2(0, 1) = (0, 1/3) \cup (2/3, 1) \subset (0, 1) = V$ the requirements are met. By writing out the equation we can find the Hausdorff dimension of the Cantor set. $1/3^s + 1/3^s = 1$, $1/3^s = 1/2$, $s = \log(2)/\log(3)$. The Cantor set is one example where the Hausdorff dimension and boxcounting dimension is equal. For a set for which Hausdorff and boxcounting dimension are equal we also call these dimensions the *fractal dimension.*

2.4 Pointwise Dimension

The pointwise dimension [5] measures the local dimension of a point on a set. Let $B_\delta(x)$ be a ball with radius δ centered at the point x. We will take the measure μ of this ball and define the pointwise dimension as follows:

$$\dim_p x = \lim_{\delta \to 0} \frac{\log \mu(B_\delta(x))}{\log \delta}.$$

If this point is part of a larger set F, we can define the average pointwise dimension by an integral:

$$\dim_p F = \int_F \dim_p(x) d_\mu(x).$$

Example 2.13. Given an interval $[0, 1]$ the measure $\mu(B_\delta(x) = \delta$ for small enough δ for any point x in this interval except for 0 and 1 where it would be $\frac{\delta}{2}$. The pointwise dimension of the interior of this interval is 1, for the points on the boundary:

$$\dim_p x = \lim_{\delta \to 0} \frac{\log \mu(B_\delta(0 \text{ or } 1))}{\log \delta} = \lim_{\delta \to 0} \frac{\log \frac{\delta}{2}\delta}{\log \delta} = \lim_{\delta \to 0} \frac{\frac{1}{2}}{\log \delta} + \frac{\log \delta}{\log \delta} = 1.$$

and we can conclude that the average pointwise dimension of $[0, 1]$ is 1;

2.5 Generalized Boxcounting Dimension

The Boxcounting dimension method tries to cover a set by boxes of equal size, if a box contains a point from the set then the box counts and otherwise not, however one can argue that one box covering more points is more important than a box containing less points. The Generalized dimension discriminates between these covering boxes.

Definition 2.14. (Generalized Dimension [5]) Let F be a non-empty bounded subset of $\mathbb{R}^n$ and let $\{B_i\}$ be a δ-cover for F. Then $P_i = \mu(B_i)/\mu(F)$ where μ is some measure. Then the generalized dimension is defined by:

$$\dim_{GD} F = \frac{1}{q-1} \lim_{\delta \to 0} \frac{\log \sum_i P_i^q}{\log \delta}.$$

Instead of looking at the amount of sets needed to cover F we look at each set in the cover and find how well it covers F on its own denoted by P. Rewriting the sum gives a weighted average $(\sum_i P_i^q)^{1/(q-1)} = (\sum_i P_i(P_i^{q-1}))^{1/(q-1)}$ where $q = 2$ is the root mean square. For $q = 0$ the generalized dimension is equal to the boxcounting dimension as $P_i^0 = 1$ and the sum over all P_i^0 would simply be the amount of sets in the δ-cover of F.

Chapter 3

Numerical Estimates

Numerical estimates arise from an approximation of a fractal where only a finite amount of points are known. This finite set of points describe the underlying fractal and thus for different algorithms we can not make the covering set as small as possible as it will then describe the dimension of the finite set of points instead of the fractal.

3.1 Correlation Dimension

The pointwise dimension is open to different numerical approaches. Suppose we want to know the pointwise dimension of F. Given $V \subset F$ a set with N points we can approximate the measure μ of $B_\delta(x)$ with $x \in V$as:

$$\mu(B_\delta(x)) \approx \frac{\#\{y \in V : y \neq x \text{ and } \|y - x\| \leq \delta\}}{N - 1}.$$

Using this we approximate the pointwise dimension at the point x as:

$$\dim_p x \approx \lim_{\delta \to 0} \lim_{N \to \infty} \frac{\log \mu(B_\delta(x))}{\log \delta}.$$

We can now find the average pointwise dimension by taking the average of $\dim_p x$ over all points but instead of doing that we calculate the average measure which we denote by $C(N, \delta)$ and find a surprisingly elegant definition of the average measure.

$$\begin{aligned} C(N, \delta) &= \frac{1}{N} \sum_{j=1}^{N} \mu(B_\delta(x_j)) = \frac{1}{N(N-1)} \#\{i \neq j \text{ and } \|x_i - x_j\| \leq \delta\} \\ &= \frac{\#\text{ of distances less than } \delta}{\#\text{ of distances altogether}} \end{aligned}$$

We define the correlation dimension as: $D_c(F) = \lim_{\delta \to 0} \lim_{N \to \infty} \frac{\log C(N, \delta)}{\log \delta}$.

Using this definition we can find an approximation of the correlation dimension by taking the limit as far as we want or can. For example: if we are given 1000 points we can fix N at 1000 and $\delta = 0.1$ to obtain an approximation of D_c. We will mostly plot D_c against δ to see how D_c approaches its limit and find how accurate our approximation is.

Given $C(N,\delta) = N_0\delta^{D_c}$ $\frac{\log N_0\delta^{D_c}}{\log\delta} = \frac{\log N_0}{\log\delta} + D_c$ the term $\frac{\log N_0}{\log\delta}$ goes to zero slowly as $\delta \to 0$. Therefore it is suggested to look at $\frac{d\log C(N,\delta)/d\delta}{d\log\delta/d\delta}$ instead while $\delta \to 0$.
This is an algorithm of order $O(N^2)$ where N is the number of points in F as we are calculating the distances between points. The parameter δ can not be chosen too high as that would lead to a higher correlation dimension and if δ is lower than the smallest interpoint distance of F then our algorithm would estimate the correlation dimension of a set of points instead of the underlying set. Our choice of δ is therefore roughly bounded between the smallest interpoint distance and the largest interpoint distance.

3.1.1 General Correlation Dimension

While the correlation dimension gives an estimation in an elegant way by the scaling between δ and the amount of distances less than δ, some details are lost, however. A concentration of points is more important than an isolated one.

$$\begin{aligned} GC_q(N,\delta) &= (\frac{1}{N}\sum_{j=1}^{N}(\mu(B_\delta(x_j)))^{q-1})^{1/(q-1)} \\ &= \left(\frac{1}{N}\sum_{j=1}^{N}\left(\frac{\#\{i \neq j \text{ and } \|x_i - x_j\| \leq \delta\}}{N-1}\right)^{q-1}\right)^{1/(q-1)}. \end{aligned}$$

$$D_{gc_q} = \lim_{\delta\to 0}\lim_{N\to\infty}\frac{\log GC_q(N,\delta,q)}{\log\delta}.$$

For $q = 2$ the general correlation dimension describes the normal correlation dimension. Mostly $q \geq 2$ as any lower will put more emphasis on sets containing less points.

3.1.2 q-point Correlation

The correlation dimension algorithm looks at the distance between two points. This can be generalized by looking at the distance between q-points.

$$\begin{aligned} C_q(N,\delta) &= \frac{1}{N^q}\#\{(x_1,...,x_q) : \|x_i - x_j\| \leq \delta, \forall i,j \in \{1,...,q\}\} \\ &= \frac{\#\text{of } q\text{-tuples within a distance of } \delta}{\#\text{of } q\text{-points}}. \end{aligned}$$

this approach has a scaling behaviour of $\lim_{N\to\infty} C_q(N,\delta) = C_q(\delta) \sim \delta^{(q-1)D_q}$

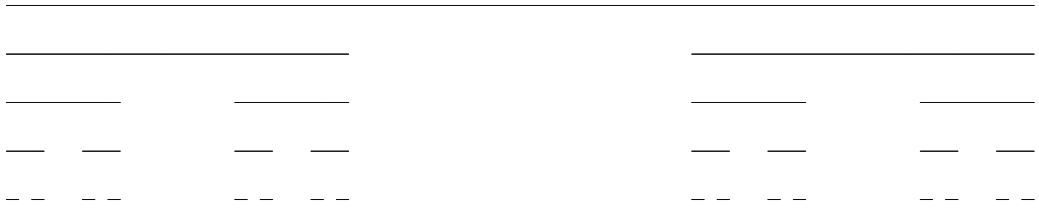


Figure 3.1: The Cantor set.

q	D_{c_q}
0	0.63071
2	0.62787
3	0.62669
4	0.62572
5	0.62499
10	0.6241
25	0.62792
100	0.63292

Table 3.1: The generalized correlation dimension obtained for different q.

$$D_{c_q} = \frac{1}{q-1} \lim_{\delta \to 0} \lim_{N \to \infty} \frac{\log C_q(N, \delta)}{\log \delta}.$$

This method can be used for any $q \geq 2$. But since the amount of q-tuples increases drasticly for $q \geq 3$ this method will lead to high estimation time for high q.

Example 3.1. As an example we will approximate the correlation dimension of the Cantor set in the interval $[0, 1]$. The Cantor set was approximated by 4096 points.
The slope of the red line is approximately $0.64 \approx \frac{\log(2)}{\log(3)}$. The dimension of the Cantor set can be approximated easily as it is self-similar meaning that as we zoom in on the set we encounter the same set multiple times, which explains the periodic behaviour that can be seen in the graphs.

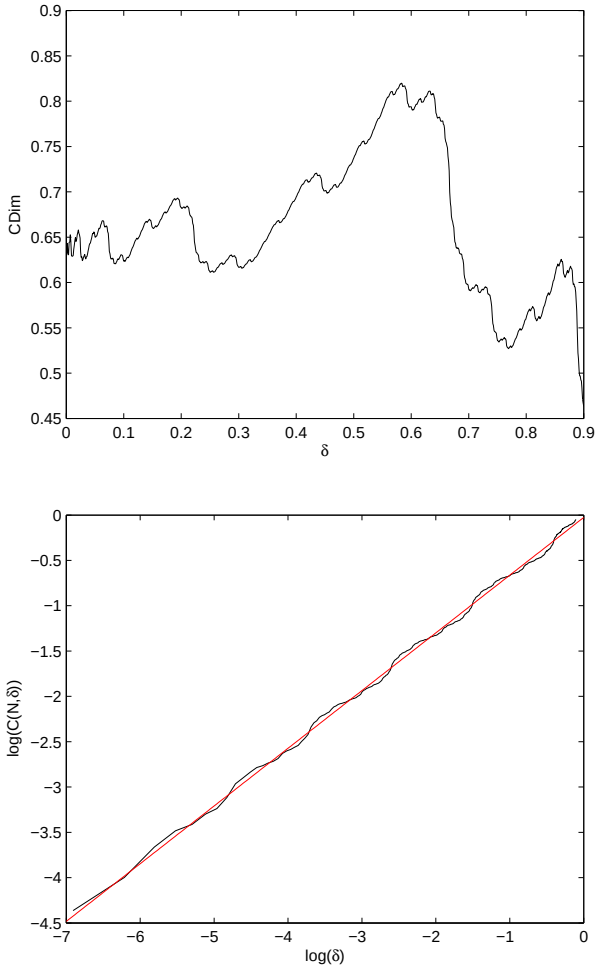


Figure 3.2: The correlation dimension of the Cantor set is approximated using two methods.

3.2 Boxcounting Algorithm

[5] We can approximate the box-counting dimension directly by breaking up $\mathbb{R}^n$ into a grid of boxes of size δ. We will then approximate $N_\delta(F)$ by counting the number of non-empty boxes. The problem with this method is that the boxcounting dimension is approached slowly as $\delta \to 0$ while we can only make δ smaller to some extent. For example, if $N_\delta(F) = N_0\delta^{-\dim_B}$ then

$$\frac{\log(N_0\delta^{-\dim_B})}{-\log\delta} = \frac{-\dim_B \log\delta + \log N_0}{-\log\delta} = \dim_B + \frac{\log N_0}{-\log\delta}$$

$\frac{\log N_0}{-\log\delta}$ decays slowly as $\delta \to 0$ thus the resolution of the fractal needs to be high in order to approximate its boxcounting dimension accurately. Instead we look at the negative slope of $\log N_\delta(F)$ versus $\log\delta$ from which we can approximate the boxcounting dimension.

$$\dim_B \approx \frac{\triangle \log N_\delta(F)}{\triangle - \log\delta}.$$

There are two possible ways in which to calculate the amount of covering boxes needed. The first algorithm applies a fixed grid onto the Julia set which covers the entire set and counts the amount of boxes that are non-empty. By using this algorithm one needs to keep track of a large amount of empty-boxes which is a strain on storage space but provides an algorithm of order $O(N)$ which is significantly faster than the correlation dimension.
The second method also applies a grid onto the Julia set but only stores non-empty boxes as it goes through all the points in F. For each point in F the algorithm checks whether it is in any non-empty box stored so far, if not a new box is created at that point. This method is slower than the last one as the amount of boxes needed to cover F converges to N as $\delta \to 0$ making this an algorithm of order $O(N^2)$ for small δ.

points	dim_B
$2^{12} = 4096$	0.64321
2^{13}	0.64125
2^{14}	0.64009
2^{15}	0.63871
2^{16}	0.63617
2^{17}	0.63694

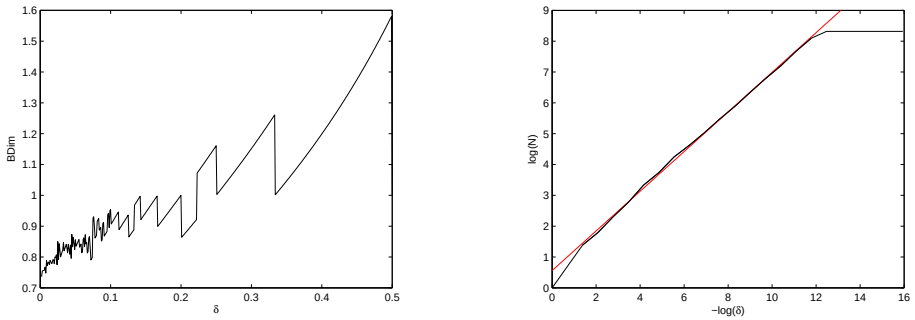


Figure 3.3: The boxcounting dimension of the Cantor set using two methods.

Example 3.2. We will estimate the boxcounting dimension of the Cantor set and compare it with the correlation dimension. Again a periodic effect is shown in the first graph as a small change in δ does not cause the amount of boxes needed to cover the set to increase. The second graph shows three different slopes, the first being too large as δ is too high and the last one is 0 as the amount of boxes is equal to the amount of points. The middle slope actually describes the boxcounting dimension of the underlying set. The slope of the red line is around 0.64321. Compared to the actual boxcounting dimension $\log 2/\log 3 \approx 0.63093$ this is a good approximation but can be improved by increasing the amount of points the Cantor set is approximated by.
Another approach is to change the starting size of the boxes. If $\delta = 1/3$ we need two boxes to cover the set and each time we divide δ by 3 the amount of boxes needed is multiplied by 2 giving us an accurate way to approach the limit: $\dim_B = \lim_{k\to\infty} \frac{\log 2^k}{-\log 1/(3^k)} = \frac{\log(2)}{\log(3)}$

3.3 Results

See appendix A for the Matlab code.

3.3.1 Cantor dust & Sierpinski triangle

The Cantor dust set is a generalization on the Cantor set and is made by removing segments from the unit plane $[0,1] \times [0,1]$ horizontally and vertically repeatedly for every plane. The Sierpinski triangle is formed from a solid triangle by removing the central triangle such that three new triangles are made congruent to the last one. This process is repeated for every new triangle. The dimension is measured by measuring the slope between $-\log(\delta)$ and $\log(N)$ taking into account that δ is not too small or high. The exact fractal dimension for the Cantor dust and Sierpinski triangle is $\frac{\log(4)}{\log(3)} \approx 1.2619$ and $\frac{\log(3)}{\log(2)} \approx 1.5849$ respectively.

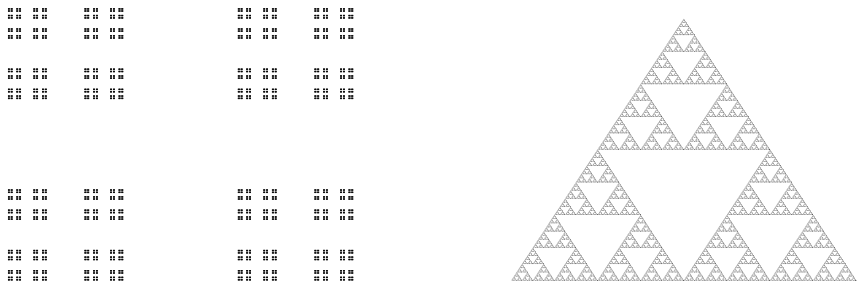


Figure 3.4: The Cantor dust set (left) and Sierpinski triangle (right)

points	$\dim_B$	$\dim_C$
4^5	1.2933	1.2927
4^6	1.2972	1.2766
4^7	1.2767	1.2693
4^8	1.2556	

points	$\dim_B$	$\dim_C$
121	1.3761	
364	1.4522	1.5698
1093	1.4958	1.552
3280	1.5211	1.5373
9841	1.5366	1.5279
29524	1.5471	
88573	1.5545	
265720	1.5595	

Table 3.2: The fractal dimension of the Cantor dust (left) and Sierpinski triangle (right).

3.3.2 Julia sets

We have already encountered the Douady rabbit set before (the Julia set for $f(z) = z^2 - 0.123 + 0.745i$), here are some results using MIIM and BSM. As before a grid will be placed on the plane where the size of each box is indicated by mesh and the maximum number of allowed points in each box is the density.

mesh	d	points	$\dim_B$
0.01	1	3286	1.360
0.001	1	81288	1.380
0.0005	1	213882	1.382
0.01	5	16900	1.372
0.001	5	422174	1.386
0.0005	5	1107556	1.387

mesh	points	$\dim_B$
0.01	3171	1.294
0.005	8424	1.349
0.001	78767	1.358
0.0005	207709	1.383

Table 3.3: The fractal dimension of the Douady rabbit using MIIM (left) and BSM (right) to generate the set.

While MIIM can plot the Douady Rabbit efficiently there are Julia sets where using MIIM will result in a "muddy" picture as the amount of detail is too high.
If the mesh size is too small then MIIM will not reveal certain parts of the set. This is

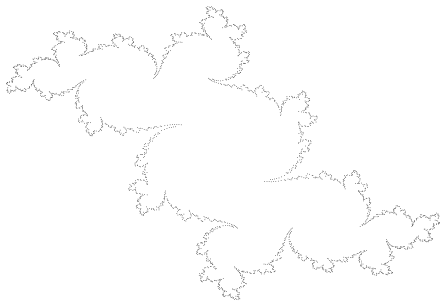


Figure 3.5: The Julia set for $c = -0.1 + 0.651i$ plotted using a mesh of 0.01.

caused by the nature of the IIM method, certain parts of the Julia set are hard to access which combined with the modification on IIM causes the stack to run out of points before the whole picture is generated. With a mesh of 0.002 the Julia set for $c = -0.1 + 0.651i$ will appear to have two "eyes" while there is supposed to be a spiral and will miss entire segments when the mesh is 0.01.

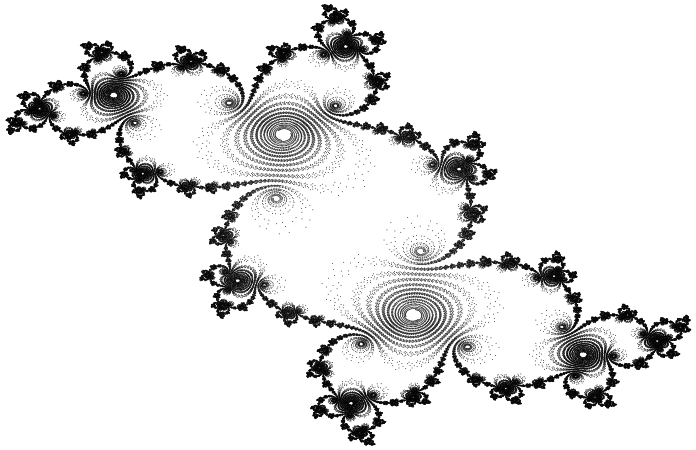


Figure 3.6: The Julia set for $c = -0.1 + 0.651i$ plotted using a mesh of 0.002.

Conclusion

For some fractals, such as the Cantor set and Douady rabbit, the fractal dimension were already found or approximated in other works and were used to estimate the performance of our algorithms. Our results are close to these previous results and we can conclude that our algorithms work sufficiently on these examples. For some Julia sets, however, the MIIM does not produce a complete image and the MBSM can not be applied on every Julia set. It is therefore important to understand the limitations of these algorithms and apply these tools correctly.

There are other dimensions we have not yet tried and other algorithms to approach them, the boxcounting dimension is one of them that is used the most. This also applies to the Julia sets as we have only used the MBSM and MIIM. For more algorithms see [7].

Appendix A

Matlab Code

A.1 Correlation dimension

In this code $r = \delta$ as in the algorithms above and F the set of points for which we want to approximate the correlation dimension. We define the values of r for which we want to calculate the correlation dimension and create the vector D which will keep track of the amount of distances $\leq r$. Then for each pair of points, without repetition, we calculate the distance and add to the vector D where neccesary. After that we can calculate the correlation dimension for each r and plot the graph in which we can find the slope. This is an algorithm of order $O(N^2)$ where N is the number of points in B.

```
N=size(F,1);

s=0.001;
max=0.5;
r=max:-s:s;
D=zeros(1,size(r,2));
for i=1:N
    i
    for j=i+1:N
        k=ceil(norm(F(i,:)-F(j,:))/s);
        if k~=0
        D(1:size(r,2)-k+1)=D(1:size(r,2)-k+1)+1;
        end
    end
end

D=2*D;

Cdim=log(D/(N*(N-1)))./log(r);
```

```
plot(log(r),log(D/(N*(N-1))),'k')
```

A.2 Boxcounting dimension

G is the grid applied to the Julia set F. For every point in F we find the box it is in and either check that box if not already checked or move on to the next point. This is done for every box size r and the result is plotted at the end. For this method we have chosen to let r decrease exponentially. This is an algorithm of order $O(N)$ where N is the number of points in F.

```
R=[];
r=1;
for l=1:12

    N=ceil(4/r);
    G=zeros(N,N);

    N=size(F,1);
    C=0;
    for i=1:N
        m=max(ceil((real(F(i))+2)/r),1);
        n=max(ceil((imag(F(i))+2)/r),1);

        if G(m,n)==0
            G(m,n)=1;

            C=C+1;
        end

    end


R=[R;r C];
r=r/2;
end

plot(R(:,1),log(R(:,2))/(-log(R(:,1))),'k')
xlabel('\delta')
ylabel('log(N)/-log(\delta)')
```

A.3 Cantor set & Cantor Dust

We start with a set $F = F_1 \cup F_2$ where $F_1\{0\}$, $F_2 = \{1\}$. At each iteration we define F_1 as the points in F multiplied by $1/3$ and F_2 as the points in F multiplied by $1/3$ and translated on the real line by adding $2/3$ to each point and finally F as the union of the two new sets. We approximate the Cantor set by applying this process a finite amount of times.

```
k=14;
F=[0; 1];

for i=1:k
    F=[F*(1/3); F*(1/3)+2/3];
end
scatter(F,zeros(size(F,1),1),1,'.','k')
```

The same method is used to generate the Cantor set is used to generate the Cantor dust.

```
k=5;
F=[0;1;1i;1+1i];

for i=1:k
    F=[F*(1/4); F*(1/4)+3/4; F*(1/4)+(3/4)*1i; F*(1/4)
      +(3/4)*(1+1i)];
end
scatter(real(F),imag(F),1,'.','k')
```

A.4 Pascal triangle

Starting with the point $(1, 0)$ we know that under the functions $f(x) = x/2$, $f(x) = (x + [1, 0])/2$, $f(x) = (x + [0.5, \sqrt{3}/2])/2$ the Pascal triangle is invariant, meaning we can generate the Pascal triangle recursively starting with one point.

```
k=7;
N=1;
for i=1:k
    N=N+3^i;
end
F=zeros(N,2);
F(1,:)=[1,0];

hold on
i=1;
N=1;
```

```
O=0;
for l=1:k

        F(N+1:N+3^l,:)=[(F(O+1:O+3^(l-1),:))/2;
        ([(F(O+1:O+3^(l-1),1))+ones(O+3^(l-1)-(O),1) (F(O+1:
          O+3^(l-1),2))])/2;
        ([(F(O+1:O+3^(l-1),1))+ones(O+3^(l-1)-(O),1)*0.5 (F(
          O+1:O+3^(l-1),2))+ones(O+3^(l-1)-(O),1)*sqrt(3)
          /2])/2];

    i=N+1;
    O=N;
    N=N+3^l;
end
 scatter(F(:,1),F(:,2),0.000001,'k')
```

A.5 IIM & MIIM

The unmodified inverse iteration method. Given a function of the form $f(z) = z^{deg} + c$ this algorithm generates an approximation of the Julia set corresponsing to $f(z)$. This algorithm can be modified for different functions by replacing the pre-image NP.

```
c=-0.123+0.765i;
deg=2;
It=15;

F=0;
I=F;
N=size(I,1);
F=[F;zeros(sum(deg.^(1:It)),1)];
Fc=1;
for k=1:It

    NI=zeros(deg^k,1);
    NIc=1;
    for j=1:N
        for v=1:1:deg
            NP=abs(I(j)-c)^(1/deg)*exp((angle(I(j)-c)/deg)
              *1i+(2*pi*1i*v)/deg);

            NI(NIc)=NP;
            NIc=NIc+1;
```

```
            Fc=Fc+1;
            F(Fc)=NP;
        end
    end

    I=NI;
    N=size(I,1);

end
scatter(real(F),imag(F),1,'k','.')
axis([-2 2 -2 2])
```

The modified IIM using a grid of boxes of size mesh and the maximum amount of points allowed in each box is the density.

```
c=-2;
deg=2;
F=1;
mesh=0.001;
density=10;
R=[-2  2 -2 2];
MaxP=400000;

F=[F; zeros(MaxP,1)];
X=zeros((R(4)-R(3))/mesh,(R(2)-R(1))/mesh);
m=max(ceil((real(F(1))-R(1))/mesh),1);
n=max(ceil((imag(F(1))-R(3))/mesh),1);
X(n,m)=X(n,m)+1;
TC=1;
TP=1;
while TP<=TC

        for v=1:1:abs(deg)

            NP=abs(B(TP)-c)^(1/deg)*exp((angle(F(TP)-c)/deg
              )*1i+(2*pi*1i*v)/deg);
            m=max(ceil((real(NP)-R(1))/mesh),1);
            n=max(ceil((imag(NP)-R(3))/mesh),1);

            if X(n,m)<density
                X(n,m)=X(n,m)+1;
                TC=TC+1;
                F(TC)=NP;
            end
```

```
        end
    TP=TP+1;

end
F=F(1:TC);
scatter(real(F),imag(F),1,'k','.')
axis(R)
```

A.6 BSM & MBSM

It is the maximum amount of iterations for each of the vertices, *mesh* is the pixel size and d the minimum distance between an iterated vertice and fixed attractive point to be labeled to a attractive basin.

```
c=-0.123+0.765i;
It=20;

mesh=0.01;
d=0.002;
R=[-2 2 -2 2];

GR=ceil((R(2)-R(1))/mesh);
GI=ceil((R(4)-R(3))/mesh);
G=ones(GI,GR);

for i=1:GI
    for j=1:GR
        i
        n=2i-mesh*i*1i;
        m=-2+mesh*j;
        At=[n+m; n+m+mesh*1i; n+m-mesh; n+m-mesh*(1-1i)];
        L=zeros(4,1);
        for x=1:It
           for z=1:size(At,1)
                if norm(At(z)-(0.022057962532054 -
                  0.031401324720234i))<d
                    L(1)=1;
                    At=[At(1:z-1);At(z+1:size(At,1))];
                    break
                elseif norm(At(z)-(-0.690855288462286 +
                  0.576387948402993i))<d
                    L(2)=1;
```

```
                    At=[At(1:z-1);At(z+1:size(At,1))];
                    break
                elseif norm(At(z)-(-0.123499489483120 +
                  0.763614701511728i))<d
                    L(3)=1;
                    At=[At(1:z-1);At(z+1:size(At,1))];
                    break
                elseif norm(At(z))>2
                    L(4)=1;
                    At=[At(1:z-1);At(z+1:size(At,1))];
                    break
                end
            end
            if sum(L)>=2 || size(At,2)==0
                break
            end
            At=At.^(2)+c;
            At=At.^(2)+c;
            At=At.^(2)+c;
        end
        if sum(L)>=2
            G(i,j)=0;
        end
    end
end
imshow(G)
```

The Modified BSM. As a seed for the algorithm to begin with, we have chosen an interval $[-2, 2]$. This algorithm generates the Douady rabbit and can be changed to generate other Julia sets by changing c and the fixed points where the edges of each pixel iterate to depending on c.

```
c=-0.123+0.765i;
It=20;
mesh=0.001;
d=0.1;
R=[-2 2 -2 2];
F=-1:mesh:1;
F=(F+mesh/2+(mesh/2)*1i)';
MaxS=1000;
MaxP=100000;

GR=ceil((R(2)-R(1))/mesh);
GI=ceil((R(4)-R(3))/mesh);
```

```
G=zeros(GI,GR);
n=max(ceil((real(F)-R(1))/mesh),1);
m=GI-max(ceil((imag(F)+-R(3))/mesh),1);
G(m,n)=1;
S=[F;zeros(size(MaxS-1,1))];
SC=size(F,1);
F=zeros(MaxP,1);
TP=0;

while size(S,2)~=0

        At=[S(1)+mesh*(1+1i); S(1)+mesh*(1-1i); S(1)+mesh
          *(-1+1i); S(1)-mesh*(-1-1i)];
        L=zeros(4,1);
            for x=1:It
                for z=1:size(At,1)
                    if norm(At(z)-(0.022057962532054 -
                      0.031401324720234i))<d
                        L(1)=1;
                        At=[At(1:z-1);At(z+1:size(At,1))];
                        break
                    elseif norm(At(z)-(-0.690855288462286 +
                       0.576387948402993i))<d
                        L(2)=1;
                        At=[At(1:z-1);At(z+1:size(At,1))];
                        break
                    elseif norm(At(z)-(-0.123499489483120 +
                       0.763614701511728i))<d
                        L(3)=1;
                        At=[At(1:z-1);At(z+1:size(At,1))];
                        break
                    elseif norm(At(z))>2
                        L(4)=1;
                        At=[At(1:z-1);At(z+1:size(At,1))];
                        break
                    end
                end
                if sum(L)>=2 || size(At,2)==0
                    break
                end
                At=At.^(2)+c;
                At=At.^(2)+c;
                At=At.^(2)+c;
```

```
            end

            if sum(L)>=2
                n=max(ceil((real(S(1))-R(1))/mesh),1);
                m=GI-max(ceil((imag(S(1))-R(3))/mesh),1);
                XG(m,n)=1;
                TP=TP+1;
                F(TP)=S(1);

                for Ra=-1:1
                    for Ia=-1:1
                        if Ia==0 && Ra==0
                            continue
                        end
                        if n+Ra>0 && n+Ra<=GR && m+Ia<=GI
                          && m+Ia>0 && G(m+Ia,n+Ra)==0
                            SC=SC+1;
                            G(m+Ia,n+Ra)=1;
                            S(SC)=S(1)+mesh*(Ra-Ia*1i);
                        end
                    end
                end
            end
            S=S(2:size(S,1));
            SC=SC-1;
end
F=F(1:TP);
scatter(real(F),imag(F),2,'k','.')
axis([-2 2 -2 2])
```

Bibliography

[1] T.G. de Jong, Dynamics of Chaotic Systems and Fractals, Bachelor Thesis in Applied Mathematics, University of Groningen, 2009.

[2] J.W. Milnor, Dynamics in one complex variable, Institute for Mathematical Sciences, SUNY, Stony Brook NY, (3-1, 3-5, 3-7, 17-2), 2006.

[3] H-O. Peitgen and P.H. Richter, The Beauty of Fractals, Springer, 1986.

[4] Kenneth Falconer, Fractal Geometry Mathematical Foundations and Applications, Second Edition, Wiley, 2003.

[5] James Theiler, Estimating fractal dimension, Lincoln laboratory, Massachussets Institute of Technology, Massachusetts 02173-9108 1989.

[6] Dietmar Saupe, Efficient Computation of Julia Sets and their Fractal Dimension, University of California, Santa Cruz, Universität Bremen, 1987.

[7] V. Drakopoulos, Comparing Rendering Methods for Julia Sets, University of Athens,

[8] D. Hilbert, Über die stetige Abbildung einer Linie auf ein Flächenstück, Mathematische Annalen, 1891.