Topological conditions on inhomogeneous fractals in Martin boundary theory and their algorithmic testing

Stefan Kohl

June 11, 2021

Abstract

We want to consider Martin boundary theory applied to inhomogeneous fractals. This is under some conditions possible, but up to now it is not clear, how one can easily check, if a certain fractal fulfills those conditions. We want to simplify one condition and further develop a computer algorithm, which can check, if an attractor fulfills the condition.

2010 Mathematics Subject Classification: 28A80, 31C35, 60J10, 60J45

Keywords: Martin boundary theory, Markov chains, Green function, fractals, multifractals.

1 Introduction

In a recent paper Freiberg and Kohl [FK19] studied the possibility to identify the attractor of a weighted iterated function system with the Martin boundary. To do so, they followed mainly the idea of Denker/Sato [DS01] and Lau/Ngai/Wang [LN12, LW15]. They adapted the transition probability of the associated Markov chain and were able to determine the Martin boundary under two conditions. The first condition, called \((B1)\), is simple and easy to verify. The second condition \((B2)\) is quite harder and it is not obvious, if a fractal fulfills \((B2)\). Because of this we want to investigate \((B2)\) to determine some properties and make this condition easier to manage.

The outline of this article is as follows: in section 2 we want to introduce the notation and summarize the work of [FK19]. We do not need all aspects of this article and thus we only present the most necessary. In section 3 we investigate \((B2)\) and reduce it in the sense that we only have to consider a finite number of words instead of an infinite number as it is in the original work. In section 4 we want to apply the ideas of section 3 to the Sierpiński gasket. Thereby we will see that we can represent the Sierpiński gasket as a Martin boundary, but sometimes not all weights of each part can be chosen arbitrarily. After this we want to consider in section 5 some facts about \((B2)\), where we introduce further the 3-level Sierpiński gasket and a more general example called \(n\)-diamond propeller.

Last but not least we develop in section 6 an algorithm to analyze every representation of a fractal. For this we investigate the Sierpiński gasket, the 3-level Sierpiński gasket, the Vicsek fractal, the Pentagasket (with and without hole) and the Hexagasket. As we will see the number of weights which can be chosen arbitrarily vary in a broad way depending on the topology of the fractal.
2 Preliminaries

Let us start with a common method of representing (and generating) fractals: iterated function systems or shorten IFS. For this, we introduce similarities \( S_i : D \subseteq \mathbb{R}^d \rightarrow D \) with \( |S_i(x) - S_i(y)| = c_i|x - y| \) and \( 0 < c_i < 1 \). An iterated function system consists of \( N \) similarities, where \( N \) has to be finite. By Hutchinsons theorem \cite{Hut81} exists an unique, non–empty compact invariant subset \( K \subset \mathbb{R}^d \) fulfilling

\[
K = S(K) := \bigcup_{i=1}^{N} S_i(K).
\]

We want to call \( K \) the attractor of the IFS and think of it as the geometric representation of the fractal. Please note, that there are multiple different IFS generating one specific fractal (respectively \( K \)).

Some of those fractals fulfill the so called open set condition, often shorten by OSC. The OSC states that there exists a non–empty bounded open set \( \mathcal{O} \subset \mathbb{R}^d \) such that \( \bigcup_{i=1}^{N} S_i(\mathcal{O}) \subset \mathcal{O} \) with the union disjoint. Since the OSC is a pre-condition of \cite{FK19}, we further want to assume that we fulfill the OSC.

Let us examine the fractal in a more accurate way. For this we introduce the alphabet \( A = \{1, \ldots, N\} \) of \( N \) letters where each letter represents one similarity. We want further to consider multiple mappings of a set. For this we introduce the word space \( \mathcal{W} \) by

\[
\mathcal{W} := \bigcup_{n \geq 1} A^n \cup \{\emptyset\},
\]

where \( \emptyset \) represents the empty word. Let us denote by \( \mathcal{W}^* \) all \( A \)-valued sequences \( w = w_1 w_2 \ldots \).

The word space \( \mathcal{W} \) itself consists of words \( w \) which can be represented by \( w = w_1 \ldots w_n \) and \( w_i \in A \). For such a word we define the length \( |w| = n \), the parent to be \( w^- = w_1 \ldots w_{n-1} \) and the restriction to the first \( m \) letters by \( w_{|m|} := w_1 \ldots w_m \) for \( m \leq |w| \). Further we define the product of two words \( v, w \in \mathcal{W} \) with \( v = v_1 \ldots v_m \) and \( w = w_1 \ldots w_n \) by \( vw = v_1 \ldots v_m w_1 \ldots w_n \).

For the empty word \( \emptyset \) we set \( |\emptyset| = 0 \) and \( w\emptyset = \emptyset w = w \) for every \( w \in \mathcal{W} \).

Let us now connect the IFS and the word space. For this we define \( S_w(E) := S_{w_1} \circ \cdots \circ S_{w_m}(E) = S_{w_1}(\cdots(S_{w_m}(E))) \) for \( E \subset \mathbb{R}^d \) and we can think of words \( w \) as cells of the fractal (and vice versa). Since some infinite sequences refer to the same point on the fractal, we should identify them as the same. In addition it is convenient to do this also on all finite words.

**Definition 2.1** \cite[Def. 2.1]{FK19}. The words \( v, w \in \mathcal{W} \) are said to be equivalent, noted by \( v \sim w \), if and only if \( |v| = |w| \), \( S_v(K) \cap S_w(K) \neq \emptyset \) and \( v^- \neq w^- \). Additionally we say, that \( v \) is equivalent to itself, such that \( v \sim v \) holds.

For \( v, w \in \mathcal{W}^* \) with \( v = v_1 v_2 \ldots \) and \( w = w_1 w_2 \ldots \) we extend this relation, such that \( v \sim w \), if and only if there exists a \( n_0 \in \mathbb{N} \) such that \( v|_{n} \sim w|_{n} \) holds for all \( n \geq n_0 \).

Further we want to define the number of equivalent words by \( R(w) := \# \{ v \in \mathcal{W} : v \sim w \} \).

Please notice, that this definition does not guarantee that this relation is indeed an equivalence relation. Luckily this is true for a big number of fractals, in particular for all nested fractals \cite[Prop. 2.9]{FK19}.

In this paper we only want to consider fractals, where \( \sim \) forms an equivalence relation (or could be modified such that \( \sim \) becomes an equivalence relation).

As a next step we want to introduce a mass distribution \( m \) on the fractal respectively on the word space. In general it would be possible to define such a mass distribution in a very general way. For our purpose we want to consider relatively simple mass distributions and use the self–similarity of our fractal. We define the mass distribution \( m \) for all \( a \in A \) in such a way that \( m(a) \in (0, 1) \) and \( \sum_{a \in A} m(a) = 1 \) holds. Further we set \( m(\emptyset) = 1 \). Now we use the self–similarity: for words \( w \in \mathcal{W} \) with \( w = w_1 \ldots w_n \) we define \( m(w) := m(w_1) \ldots m(w_n) \).

If all those weights \( m(i) \) are chosen equal (i.e. \( m(i) = \frac{1}{N} \)) we say that we consider the homogeneous
We want to introduce a Markov chain on $\mathcal{W}$. For this we adopt [FK19, Def. 3.1] and define the transition probability $p : \mathcal{W} \times \mathcal{W} \to [0, 1]$ by

$$p(v, w) := \begin{cases} \frac{m(w)}{\sum_{\hat{v} \sim v} m(\hat{v})}, & \text{if } w = \hat{v}i \text{ with } \hat{v} \sim v \text{ and } i \in \mathcal{A}, \\ 0, & \text{else.} \end{cases}$$

Further we introduce the $n$-step transition probability $p_n : \mathcal{W} \times \mathcal{W} \to [0, 1]$ recursively by

$$p_n(v, w) := \begin{cases} \delta_v(w), & n = 0, \\ \sum_{u \in \mathcal{W}} p_{n-1}(v, u) p(u, w), & n \geq 1. \end{cases}$$

Since $p_n(v, w)$ is only positive for a specific $n$ we define the Green function $g : \mathcal{W} \times \mathcal{W} \to \mathbb{R}$ by

$$g(v, w) := \sum_{n=0}^{\infty} p_n(v, w) = p|w|_\sim(v, w), \quad v, w \in \mathcal{W}.$$  

The Martin kernel $k : \mathcal{W} \times \mathcal{W} \to \mathbb{R}$ is then defined as a sort of “renormalized” Green function, namely by

$$k(v, w) := \frac{g(v, w)}{g(\emptyset, w)} = \frac{g(v, w)}{m(w)}, \quad v, w \in \mathcal{W},$$

where we used in the second identity Theorem 3.7 from [FK19], which states that $g(\emptyset, w) = m(w)$ for all $w \in \mathcal{W}$.

Based on the Martin kernel one can define the Martin metric $\rho$, which exact form is for now irrelevant. The Martin space $\overline{\mathcal{W}}$ is then the $\rho$-completion of $\mathcal{W}$ and the Martin boundary $\mathcal{M}$ is the boundary of $\overline{\mathcal{W}}$:

$$\mathcal{M} := \overline{\mathcal{W}} \setminus \mathcal{W}.$$  

This Martin boundary is in some sense connected to the fractal and the attractor. For the homogeneous case this was first studied by [DS01, DS02] and later in [JLW12, LN12, LW15]. Under the following conditions this can also be done in the inhomogeneous case.

**Assumption (FK19).** We make the following assumptions:

(A) The relation $\sim$ is an equivalence relation

(B1) The Martin kernel in the homogeneous case exists.

(B2) For all $w \in \mathcal{W}$ holds either

$$m(w) = m(\hat{w}) \quad \forall \hat{w} \sim w$$

or

$$w \sim (\hat{w})^{-} \quad \forall \hat{w} \sim w.$$  

With this we are able to identify the attractor $K$ with the inhomogeneous Martin boundary:

**Theorem 2.2 (FK19 Theorem 5.4 / Cor. 5.5).** Under (A), (B1) and (B2) the inhomogeneous Martin boundary coincides with the homogeneous Martin boundary and

$$K \simeq \mathcal{W}^\sim/\sim \simeq \mathcal{M}_\text{hom} = \mathcal{M}.$$  

This result is stunning. At the same time it arises the question, when the preconditions are fulfilled. Condition (B1) can be easily checked, for example we can use the results of [LW15]. Condition (B2) is quite harder for this reason and in the following we want to study (B2) in the following intensely.
3 Simplifying condition \((\text{B2})\)

In a first step we want to examine how we can express the condition \((\text{B2})\) in an easier way. This should also help to detect iterated function systems where \((\text{B2})\) is in the homogeneous case fulfilled. In other words, we want to reverse the problem and analyze which conditions imply \((\text{B2})\).

For this we formulate our first helpful lemma.

**Lemma 3.1.** Let \(\tilde{v} = uv \in W\) with \(u,v \in W\) and \(u \neq \emptyset\). If all \(\tilde{w} \sim \tilde{v}\) can be expressed as \(\tilde{w} = uw\) and \(v \sim w\) fulfills \((\text{B2})\), then \(\tilde{v} \sim \tilde{w}\) fulfills also \((\text{B2})\).

**Proof.** Consider \(\tilde{v} = uw \in W\) with \(u,v \in W\) and \(u \neq \emptyset\). Suppose that we can express all equivalent words \(\tilde{w} \sim \tilde{v}\) by \(\tilde{w} = uw\). Further \(v \sim w\) satisfies \((\text{B2})\). We want to distinguish two cases, depending on which condition of \((\text{B2})\) is fulfilled by \(v \sim w\).

In the first case we consider that \(v \sim w\) fulfills \((\text{B2})\) by \(m(v) = m(w)\). This implies that

\[m(\tilde{v}) = m(uv) = m(u)m(v) = m(u)m(w) = m(uv) = m(\tilde{w})\]

holds. Thus \(\tilde{v} \sim \tilde{w}\) fulfills \((\text{B2})\) by the first condition.

Consider as the second case that \(v \sim w\) fulfills \((\text{B2})\) by \(v^- \sim w^-\). We claim that \(\tilde{v}^- \sim \tilde{w}^-\) holds.

To verify this we consider the definition of the equivalence relation. Both words \(\tilde{v}^-\) and \(\tilde{w}^-\) have the same length since \(|\tilde{v}| = |\tilde{w}|\) holds. By \(v \sim w\) follows \(v^- \neq w^-\) and since \(v^- \sim w^-\) must hold we get

\[S_{v^-}(\tilde{v}) \cap S_{w^-}(\tilde{w}) \neq \emptyset. \quad (1)\]

If we apply \(S_u(\cdot)\) to \(1\) we obtain \(S_{v^-}(\tilde{v}) \cap S_{w^-}(\tilde{v}) \neq \emptyset\).

As last part it remains that \((\tilde{v}^-)^- \neq (\tilde{w}^-)^-\) holds. By \(v^- \sim w^-\) and \(v^- \neq w^-\) follows \((v^-)^- \neq (w^-)^-\). This implies \((w^-)^- \neq (w^-)^-\) respectively \((\tilde{v}^-)^- \neq (\tilde{w}^-)^-\).

Thus we receive that \(\tilde{v}^- \sim \tilde{w}^-\) holds which implies that \(\tilde{v} \sim \tilde{w}\) fulfills \((\text{B2})\) by the second condition and we can complete the proof. \(\square\)

As a next step we want to distinguish between two cases, one which fulfills \((\text{B2})\) automatically and one where we can find an alternative formulation of \((\text{B2})\).

**Corollary 3.2.** Let \(v = v_0v_1 \ldots v_n \in W\). If \(v_0 \neq w_0\) holds for all \(w \sim v\) \((w = w_0w_1 \ldots w_n)\) and \(v\) should satisfy \((\text{B2})\), then:

- if one \(w \sim v\) fulfill \(w^- \not\sim v^-\): \(v\) and all \(w\) must fulfill \(m(v) = m(w)\),
- if all \(w \sim v\) fulfill \(w^- \sim v^-\): \((\text{B2})\) is automatically fulfilled.

**Proof.** By Lemma 3.1 we only have to consider words which differ already in the first letter. Thus we want to consider \(v \in W\) and all equivalent words \(w \sim v\) fulfilling \(w_0 \neq v_0\). Further \((\text{B2})\) should hold.

In the first case there exists an equivalent word \(w\) with \(w^- \not\sim v^-\). Since \((\text{B2})\) must hold, the first condition must be fulfilled and thus

\[m(w) = m(v) \quad \text{for all } w \sim v.\]

In all other cases all equivalent words \(w\) fulfill \(w^- \sim v^-\). Immediately it follows that \((\text{B2})\) is fulfilled by the second condition. \(\square\)

We can combine the previous lemma and corollary into one single statement, which specify when \((\text{B2})\) is fulfilled.

**Proposition 3.3.** Consider \(v = v_0 \ldots v_n\) and \(w = w_0 \ldots w_n\). If all relations \(v \sim w\) with \(v_0 \neq w_0\) and \(v^- \not\sim w^-\) fulfill \(m(v) = m(w)\) for all \(\tilde{w} \sim v\), then \((\text{B2})\) is satisfied.

**Proof.** Let \(v, w \in W\). Lemma 3.1 implies that we only have to consider relations where the first letter differs. If further the parents \(v^-, w^-\) are equivalent for all \(w \sim v\) we can apply the second case of Corollary 3.2 and \((\text{B2})\) is fulfilled.

If there exists one equivalent word \(w \sim v\) which parent is not equivalent to \(v^-,\) but at the same time \(m(\tilde{w}) = m(v)\) holds for all \(\tilde{w} \sim v\), \((\text{B2})\) is satisfied by its second condition. \(\square\)
Figure 1: Generating the Sierpiński gasket in a descriptive way

(a) The standard mapping with no rotation and no flipping
(b) No rotation, but flipping.
(c) Rotate the triangle by \(-\frac{2}{3}\pi\) without flipping.
(d) Rotate the triangle by \(-\frac{2}{3}\pi\) and flip it over.
(e) Rotate the triangle by \(\frac{2}{3}\pi\) without flipping.
(f) Rotate the triangle by \(\frac{2}{3}\pi\), further flip it.

Figure 2: 6 possible mappings for the smaller copies at the Sierpiński gasket. The coloring should help to understand the orientation and the flipping.

We want to use this proposition extensive in the following, since it allows us to check only a small amount of words in \(W\). In contrast to this we would have to check all words in \(W\), if we want to verify \([B2]\) in a direct way. This is indeed very problematic, since we would have to check infinite many words. In other words Proposition 3.3 allows us to check only finite many words because of the self-similarity. As we will see this amount of words is very low. For example we only have to check 3 relations on the Sierpiński gasket, which we will see in the next section.

4 Sierpiński gasket as the simplest example

We want to apply and understand the results from section 3. For this, we consider one of the simplest fractals one can think of: the Sierpiński gasket. To be more precise, we want to consider the Sierpiński gasket in the plane, shorten by SG. The SG is generated through three smaller copies of a triangle, which are arranged such that they form again the starting triangle with a hole. This is done infinite times and the resulting figure is called Sierpiński gasket. Figure 1 should help to understand this procedure. Of course one could also specify the three similarities in a proper mathematical way, but this would only distract ourselves from the problem, since it is more a topological problem. Instead we want to analyze in which way each copy can be arranged. So let us consider on single small copy. This copy can be rotated by 0, \(\frac{2}{3}\pi\) or \(-\frac{2}{3}\pi\) and may be flipped. In combination we get six possible ways to map the big triangle onto the small copy. Those six ways are also illustrated in figure 2.

We can apply one of the six mappings independently on all three copies, thus we get in total \(6^3 = 216\) possible IFS. This number initially appears small. But if we consider other fractals the number of possible IFS increases rapidly. Because of this, we should consider all IFS in a general way.

For the Sierpiński gasket we want to write down the general word space. This can be seen in Figure 3 for words of length 2. We choose the coding in such a way that the first letter is fixed and thus the rotation and flipping does not matter for the first letter. For the second letter they come into account and the second letter has to variable. We can denote the children of 1 by \(1a_i\) and similar the children of 2 (resp. 3) by \(b_i\) (\(c_i\)). It must hold, that \(a_1, a_2, a_3 \in \{1, 2, 3\}\) and are pairwise distract. The same must hold for \(b_i\) respectively \(c_i\). By the definition of the equivalence relation it follows, that \(1a_2 \sim 2b_1, 1a_3 \sim 3c_1\) and \(2b_3 \sim 3c_2\) must hold. By Proposition 5.3 it is
Figure 3: The general word space of the Sierpiński gasket. It must hold, that $a_i$ (resp. $b_i$ and $c_i$) are pairwise different and $a_i, b_i, c_i \in \{1, 2, 3\}$.

![Iterated function system](image)

(a) Iterated function system

![Word space W](image)

(b) Word space $W$

Figure 4: An IFS where no weights can be chosen. The coloring should help to understand the orientation of the contractions. It must hold, that $m(1) = m(2) = m(3)$ and thus $m(a) = \frac{1}{3}$ for $a \in A$.

sufficient, to consider only those three relations. Thus, if $[B2]$ should be fulfilled, the equations

$$
\begin{align*}
    m(1a_2) &= m(2b_1), \\
    m(1a_3) &= m(3c_1), \\
    m(2b_3) &= m(3c_2)
\end{align*}
$$

must hold. Remember, that $m$ is multiplicative, so $m(1a_2) = m(1)m(a_2)$. Since $m$ is a mass distribution, we have to add

$$
\begin{align*}
    m(1) + m(2) + m(3) &= 1, \\
    m(a) &> 0, a \in A
\end{align*}
$$

to the equations in $[2]$. In total we get four algebraic equations in three positive variables $m(1)$, $m(2)$ and $m(3)$. The only problem are the values of $a_1, \ldots, c_3$, which differ with each IFS. For this reason we want consider in the following three different examples and we will get a deeper insight in the dependency of choosing small mappings fulfilling $[2]$ and $[3]$.

**Example 4.1.** Consider the IFS with $a_1 = 1, a_2 = 2, a_3 = 3, b_1 = 3, b_2 = 1, b_3 = 2, c_1 = 2, c_2 = 3, c_3 = 1$. Figure 4 shows this IFS in a graphical way, moreover the word space is indicated. The equations $[2]$ and $[3]$ turn into

$$
\begin{align*}
    m(1)m(1) &= m(2)m(3), \\
    m(1)m(3) &= m(3)m(2), \\
    m(2)m(2) &= m(3)m(2), \\
    m(1) + m(2) + m(3) &= 1.
\end{align*}
$$

Those equations can easily be solved (unlike the general equations in $[2]$). We get, that $m(1) = m(2) = m(3) = \frac{1}{3}$ must hold. We conclude therefore, that this IFS only fulfills $[B2]$ in the homogeneous case, which is (for us) relatively uninteresting.

The next example is much more interesting, since we get a first weighted example of the Sierpiński gasket fulfilling $[B2]$.
Example 4.2. Let \( a_1 = 1, a_2 = 2, a_3 = 3, b_1 = 1, b_2 = 2, b_3 = 3, c_1 = 2, c_2 = 1, c_3 = 3 \). This iterated function system and the word space are illustrated in Figure 5.

The corresponding equations, which need to be fulfilled in order to fulfill \((B2)\) are then

\[
\begin{align*}
m(1)m(2) &= m(2)m(1), \\
m(1)m(3) &= m(3)m(1), \\
m(2)m(3) &= m(3)m(1), \\
m(1) + m(2) + m(3) &= 1.
\end{align*}
\]

Immediately one can see that \( m(1) = m(2) \) must hold. At the same time the equations make no restrictions on \( m(3) \), except from \( 2m(1) + m(3) = 1 \). This implies that we either can choose \( m(3) \in (0,1) \) with \( m(1) = m(2) = \frac{1-m(3)}{2} \) or we can choose \( m(1) \in (0,\frac{1}{2}) \) with \( m(3) = 1 - 2m(1) \) (and of course \( m(2) = m(1) \)).

In both cases we can choose the weights such that they are inhomogeneous and at the same time \((B2)\) is fulfilled. Since we can choose one weight, we want to call this IFS a case with one free weight or one free parameter.

The previous examples show that it depends on the IFS, if \((B2)\) can be fulfilled in the inhomogeneous case. The question arises whether \((B2)\) is fulfilled and we can choose more than one weight arbitrarily. The next example should help to clear this question.

Example 4.3. Consider the original Sierpiński gasket without rotations or flippings (see Figure 3). In this case we have \( a_1 = 1, a_2 = 2, a_3 = 3, b_1 = 1, b_2 = 2, b_3 = 3, c_1 = 1, c_2 = 2 \) and \( c_3 = 3 \) with

\[
\begin{align*}
m(1)m(2) &= m(2)m(1), \\
m(1)m(3) &= m(3)m(1), \\
m(2)m(3) &= m(3)m(2), \\
m(1) + m(2) + m(3) &= 1.
\end{align*}
\]

The first three equations are automatically fulfilled and only \( m(1) + m(2) + m(3) = 1 \) remains. Thus we can choose two weights and the third is determined by this equation. For example we can choose first \( m(1) \in (0,1) \). Then we choose \( m(2) \in (0,1-m(1)) \), where the upper bound of \( 1-m(1) \) comes from the fact, that \( m(1) + m(2) < 1 \) must hold because of \( m(3) > 0 \). We can proceed in the same way, if we want to choose other weights.

In total we have two free parameters on this particular IFS.

Those three examples show that it purely depends on the topology of each fractal. In other words, we can consider a certain fractal and may look at all iterated function systems which generate this fractal. Depending on the topology we get a different number of free parameters. If we do this for the remaining 213 IFS of the Sierpiński gasket we get that there are in total 194 IFS with zero free
parameters, 21 IFS with one free parameter and exactly one (!) IFS with two free parameters, the one from Example 4.3.

Because of this, we want to establish in section 6 an algorithm which analyzes all possible IFS to a certain fractal and returns the number of free parameters. But before we do this, let us first take a closer look at some facts about (B2).

5 Some general facts

We want to collect in this section some facts about (inhomogeneous) fractals and the condition (B2). As a first topic we want to answer the question if there is a minimal inhomogeneous fractal which fulfills (B2). Later we want to set up a minimal example which won’t fulfill (B2). Those minimal examples should fulfill some conditions such that they are actually real minimal examples:

- We want the number of copies resp. the length of the alphabet $\mathcal{A}$ to be minimal.
- The open set condition should hold. This condition seems to be not relevant for finding a minimal example, since we could consider the so called Haka Tree (for more information see [Kig01, Example 1.2.9]. But since we want to apply the results of [FK19] we need the OSC.
- All those examples should be indeed fractals (see for example [Fal90, Introduction] for a definition of a fractal through a list of properties). Of course, one could define the interval $[0, 1]$ as a self–similar fractal with two (or more copies), but it is obvious that this is only an artificial fractal.
- Those examples should contain at least a word $w$ with $R(w) > 1$. If we would consider a fractal where all words $w$ fulfill $R(w) = 1$, then (B2) would be fulfilled automatically. This would make the minimal example meaningless.
- The mass distribution should be inhomogeneous, otherwise (B2) is also automatically fulfilled.

In other words, our minimal example should be an inhomogeneous fractal, minimal in the sense of the number of copies and should contain a word $w$ with $R(w) > 1$.

**Example 5.1** (Minimal inhomogeneous fractal fulfilling (B2)). Our minimal example needs at least two small copies, otherwise we are unable to define an IFS. In fact this is possible with the von Koch curve, which is named after the Swedish mathematician Helge von Koch (1870 – 1924) who introduced this fractal in 1904 [Koc04].

The typical IFS which generates the von Koch curve uses normally four small copies and arranges them in such a way that the small copies form the typical tent–form. Besides this there is another...
IFS which has the von Koch curve as attractor. This IFS consists of the two mappings

\[
S_1 : \mathbb{R}^2 \to \mathbb{R}^2, y \mapsto \frac{1}{\sqrt{3}} \begin{pmatrix} -\cos \left( \frac{7\pi}{6} \right) & \sin \left( \frac{7\pi}{6} \right) \\ -\sin \left( \frac{7\pi}{6} \right) & \cos \left( \frac{7\pi}{6} \right) \end{pmatrix} y = \\
\quad \frac{1}{\sqrt{3}} \begin{pmatrix} \frac{\sqrt{3}}{2} & 1 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{pmatrix} y,
\]

\[
S_2 : \mathbb{R}^2 \to \mathbb{R}^2, y \mapsto \frac{1}{\sqrt{3}} \begin{pmatrix} -\cos \left( \frac{5\pi}{6} \right) & \sin \left( \frac{5\pi}{6} \right) \\ -\sin \left( \frac{5\pi}{6} \right) & \cos \left( \frac{5\pi}{6} \right) \end{pmatrix} y + \left( \frac{1}{\sqrt{3}} \right) = \\
\quad \frac{1}{\sqrt{3}} \begin{pmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \end{pmatrix} y + \left( \frac{1}{\sqrt{3}} \right).
\]

The two mappings are also presented in Figure 7a in a graphical way, which might be much easier to understand. The coloring of the mappings should help to understand how the small copies are arranged. Figure 7 contains further a second figure. The Figure 7b shows the word space up to length 2 and further the attractor of the IFS.

The alphabet consists obviously of \( A = \{1, 2\} \) and the word space of all words \( w = w_1w_2 \ldots w_n \) with \( w_i \in A \). We apply Proposition 3.3 and hence the equation

\[
m(12) = m(21) \tag{4}
\]

has to be fulfilled. Since \( m \) is multiplicative, it follows that (4) is always fulfilled. So, the only restriction is the fact that \( m \) has to be a mass distribution with

\[
m(1) + m(2) = 1.
\]

Therefore it follows that we can choose \( m(1) \in (0, 1) \) and \( m(2) = 1 - m(1) \). Of course we could also choose \( m(2) \in (0, 1) \) and determine \( m(1) \).

Thus we found an inhomogeneous fractal which fulfills (B2). The attractor is also a “real” fractal, since the Hausdorff dimension equals \( \frac{\ln 4}{\ln 3} \approx 1.262 \). We also note that this fractal is p.c.f.

As a logical consequence we can ask for a minimal example which won’t fulfill (B2).

**Example 5.2** (Minimal inhomogeneous fractal not fulfilling (B2). For this minimal counter-example we use again the von Koch curve, which we already introduced in the previous Example 5.1. For the counter-example we arrange the small copies in a different way. We choose the
mappings as

\[
S_1 : \mathbb{R}^2 \to \mathbb{R}^2, \quad y \mapsto -\frac{1}{\sqrt{3}} \left( \frac{\sqrt{3}}{2} \cos \left( \frac{\pi}{6} \right) - \sin \left( \frac{\pi}{6} \right) \right) y + \left( \frac{1}{2} \frac{\sqrt{3}}{6} \right) = \\
-\frac{1}{\sqrt{3}} \left( \frac{\sqrt{3}}{2} - \frac{1}{2} \frac{\sqrt{3}}{2} \right) y + \left( \frac{1}{2} \frac{\sqrt{3}}{6} \right)
\]

\[
S_2 : \mathbb{R}^2 \to \mathbb{R}^2, \quad y \mapsto -\frac{1}{\sqrt{3}} \left( \frac{\sqrt{3}}{2} \cos \left( -\frac{\pi}{6} \right) - \sin \left( -\frac{\pi}{6} \right) \right) y + \left( \frac{1}{0} \right) = \\
-\frac{1}{\sqrt{3}} \left( -\frac{\sqrt{3}}{2} + \frac{1}{2} \frac{\sqrt{3}}{2} \right) y + \left( \frac{1}{0} \right)
\]

which is also illustrated in Figure 8a. The second Figure 8b contains the corresponding word space and again the attractor. The intention of this IFS is the fact that the cells “11” and “22” touch each other and are therefore equivalent.

By Proposition 3.3, the equation

\[
m(11) = m(22)
\]

has to be fulfilled. For \(m(1) > 0\) and \(m(2) > 0\) this is only fulfilled in the case of \(m(1) = m(2)\). This means that an inhomogeneous mass distribution on this IFS can not fulfill (B2).

This example is a valid minimal example using the same arguments as in Example 5.1 and is also p.c.f..

In total we get two examples, which fulfill our criteria of a minimal example. Thereby, one example fulfills (B2) and the other example does not fulfill (B2).

As a next step we want to consider so called nested fractals, which are characterized as follows:

**Definition 5.3 (Ham00).** We want to denote by \(F_0' := \{ q_i : S_i(q_i) = q_i \} \) the set of all fixed points of the similarities \(S_i\). Further we want to define the set of all essential fixed points \(F_0\) by \(F_0 := \{ x \in F_0' : \exists i, j \in A, y \in F_0', x \neq y \text{ st. } S_i(x) = \bar{S}_j(y) \}\). A fractal \(K\) is then called nested, if it satisfies:

1. **Connectivity:** For any 1-cells \(C_i, \cdots, C_n\) of 1-cells such that \(C_0 = C_n = C'\) and \(C_{i-1} \cap C_i \neq \emptyset, i = 1, \ldots, n\).
2. **Symmetry:** If \(x, y \in F_0\), then reflection in the hyperplane \(H_{xy} = \{ Z : |z - x| = |z - y| \}\) maps \(S^n(F_0)\) to itself.
3. **Nesting:** If \(v, w \in W\) with \(v \neq w\), then \(S_v(K) \cap S_w(K) = S_v(F_0) \cap S_w(F_0)\)
Figure 9: The 3-level Sierpiński gasket, which is generated in the same way as the Sierpiński gasket, but with 6 smaller copies. This SG₃ has no rotations or flippings.

4. Open set condition OSC: There is a non-empty, bounded, open set $O$ such that the $S_i(O)$ are disjoint and $\bigcup_{i=1}^{N} S_i(O) \subseteq O$.

Those nested fractals form a class of fractals with some nice properties. One of those properties is the following: if the fractal is nested, then the equivalence relation is indeed equivalent [FK19, Prop. 2.9]. Since we only want to consider fractals with such an equivalence relation, this is a nice pre-condition.

Unfortunately, the property “nested” does not imply that $[B2]$ is fulfilled in the inhomogeneous case, which shows the next example.

**Example 5.4.** Let us consider the so called 3-level Sierpiński gasket, shorten by SG₃. This is a modification of the normal Sierpiński gasket and consists of six copies with contraction ratios $\frac{1}{3}$. Those are arranged in such a way that they form again a triangle. In Figure 9 this is also visualized. Again, we could flip and rotate the smaller copies, but for now we want to consider the case with no flipping or rotating. The associated word space is included in Figure 9. It holds that in this case the SG₃ is nested, where $F_0$ consists of the edges of the starting triangle. By Proposition 5.3 we only have to consider the relations $12 \sim 41$, $13 \sim 51$, $21 \sim 42$, $23 \sim 62$, $31 \sim 53$, $32 \sim 63$, $43 \sim 52 \sim 61$. We can set up all needed equations and become

$$m(12) = m(41)$$
$$m(13) = m(51)$$
$$m(21) = m(42)$$
$$m(23) = m(62)$$
$$m(31) = m(53)$$
$$m(32) = m(63)$$
$$m(43) = m(52) = m(61)$$
$$m(1) + m(2) + m(3) + m(4) + m(5) + m(6) = 1$$

We can apply the fact that $m(ab) = m(a)m(b)$ holds, shorten the equations and receive

$$m(2) = m(4)$$
$$m(3) = m(5)$$
$$m(1) = m(4)$$
$$m(3) = m(6)$$
$$m(1) = m(5)$$
$$m(2) = m(6)$$
$$m(43) = m(52) = m(61)$$
$$m(1) + m(2) + m(3) + m(4) + m(5) + m(6) = 1.$$ 

As we can see, it must hold that $m(a) = \frac{1}{6}$ for $a \in A$, which implies that we are not able to choose an inhomogeneous mass distribution and at the same time $[B2]$ is fulfilled.

Thus this is a suitable counter example for the fact, that the property “nested” cannot imply the fact that $[B2]$ is fulfilled in the inhomogeneous case.

In the following we want to show, that $[B2]$ is not to restrictive. In particular we want to consider inhomogeneous fractals, which have words $\hat{w}$ with $R(\hat{w}) > 2$. We split this into two examples,
Example 5.5. For an example with \( R(\hat{w}) = 3 \) we can use the 3-level Sierpiński gasket, which we already introduced in the previous Example 5.4. In contrast to this example we want to rotate some copies, but won’t flip. If we rotate \( S_2 \) and \( S_3 \) by \(-\frac{2\pi}{3}\) and \( S_6 \) by \( \frac{2\pi}{3}\) we get a different word space, which is illustrated in Figure 10. As a first observation we note, that \( \sim \) forms still an equivalence relation. Further we can choose the mass of the cell 1 with \( m(1) \in (0,1) \). For all other cells it holds, that \( m(a) = 1 - m(1) \) for \( a \in \{2, \ldots, 6\} \).

It then holds, that \( \{B2\} \) is fulfilled for all \( w \in W \). For this, we can analyze words of length 2, which will not fulfill the property \( \{B2\} \) by \( w^- \sim (\hat{w})^- \). Thus they must fulfill \( m(w) = m(\hat{w}) \) for all \( \hat{w} \sim w \), which shows the following list:

\[
\begin{align*}
m(12) &= m(41) \\
m(42) &= m(22) \\
m(21) &= m(61) \\
m(62) &= m(33) \\
m(32) &= m(53) \\
m(51) &= m(13) \\
m(43) &= m(52) = m(63)
\end{align*}
\]

For a word of length greater 2 we can differ between two cases. Either \( w \) fulfills \( w^- \sim (\hat{w})^- \) for all \( \hat{w} \sim w \). In this case \( \{B2\} \) is fulfilled. In the other case \( w \) can be expressed as \( w = uab \) and \( \hat{w} = ucd \) with \( u \in W \) and \( a, b, c, d \in W \). It then holds, that we can apply the results for words of length 2, where \( m(ab) = m(cd) \) holds. Thus \( m(w) = m(uab) = m(ucd) = m(\hat{w}) \) follows and \( \{B2\} \) is fulfilled for all \( w \in W \).

For now it could be possible, that this example can be modified in such a way, that we can choose the mass of two cells arbitrarily. As we will later see in section 6, there is no way of generating the SG\(_3\) with six copies and two or more free parameters.

Example 5.6 (n-diamond propeller). The previous Example 5.5 already gives a good idea, how we could construct an example (in \( \mathbb{R}^2 \)) with a cell \( \hat{w} \) and \( R(\hat{w}) = n \). Basically, we just need \( n \) touching copies of the fractal in one single point \( p \). We can achieve this, if we choose our basic form to be a diamond and we want to call this example the \( n \)-diamond propeller.

Since we do not want overlapping copies the angle in one corner of the diamond must be sufficiently small. For now, we want to denote this angle by \( \hat{\alpha} \). We notate it with a tilde, since we have to modify this angle and denote later the final angle by \( \alpha \).

Let us fix \( n \geq 4 \) as the number of touching copies in \( p \). If we choose \( \hat{\alpha} \leq \frac{\pi}{n} \), we can arrange the copies in such a way, that they only touch in \( p \). As it turns out, it is even more practically, if we...
Our aim is that this fractal fulfills (B2) and the mass distribution is inhomogeneous. If we choose \( \alpha \) small enough, arranging \( n \) smaller copies with \( \alpha \) small enough guarantees that
\[ \sim \]
\( w \)
such a cell immediately.

In the first case we get that all copies intersect in \( p \) and thus \( \sim \) is transitive.

We can quickly notice, that Figure 12 should help to understand this word space.

Let us now consider the word space. For this, we denote the cells in line from top to bottom by \( 1, \ldots, n \),

with \( n_0 := \frac{1}{c} \).

The remaining cells arranged in a circle around \( p \) should be denoted by
\[ n_0 + 1, \ldots, N \]
where \( N = n_0 - 2 + n \) holds. We want to orientate the copies from top to bottom in such a way, that two neighboring cells \( u \) and \( v \) touch in \( u1 \) and \( v1 \) respectively \( u_n \) and \( v_n \), except the cell \( n_0 \). The cell \( n_0 \) is not rotated and it should hold, that \( n_01 \) intersects with the cell \( (n_0 - 1)n_0 \).

Since \( n_0 \) is a multiple of four, it holds, that the cells \( u_{n_0} \) and \( (\frac{1}{n_0} + 1) \) meet exactly in \( p \). Moreover their orientation is in the way, that they intersect in \( (\frac{1}{n_0}1) \) and \( (\frac{1}{n_0} + 1)1 \). The remaining copies are accumulated around \( p \). For such a copy \( w \) it should hold that \( w1 \) intersects with the other cells.

Figure 12 should help to understand this word space.

We can quickly notice, that \( \sim \) forms an equivalence relation since there are only two types of touching cells. Either the cells touch in \( p \) (or iterations of it) or they touch in the line from top to bottom (and also in iterations of it). Both cases are harmless. For simplicity we only want to consider cells with word length 2.

In the first case we get that all copies intersect in \( p \) and thus \( \sim \) is transitive.

Further we have for such a cell \( w \) the property \( R(w) = n \). In the other case we only have cells with \( R(w) = 2 \), which immediately guarantees that \( \sim \) is transitive, thus \( \sim \) is an equivalence relation.

Our aim is that this fractal fulfills (B2) and the mass distribution is inhomogeneous. If we choose

\[
\alpha_0 := \pi \left( 4 \left\lceil \frac{n}{4} \right\rceil \right)^{-1} \leq \tilde{a},
\]

where \( \lceil \cdot \rceil \) is the ceiling function.

This allows us, that four copies form a “+” in \( p \). We can then choose the contraction ratio \( c_0 \) of all similarities as
\[
c_0 := \frac{1}{2} \tan \left( \frac{\alpha_0}{2} \right).
\]

This implies, that the smaller copies just fit horizontal in the original diamond.

In the next step we add at the top and the bottom of the diamond two additional copies, which guarantees us, that the shape of the OSC–set \( O \) is (at least) diamond–like.

In the last construction step we want to connect those outer cells with the inner cells. We want to avoid copies with improper rotations, instead they should form a proper line. For this, we have to choose the contraction ratio smaller and it should be a multiple of four, such that we set the new contraction ratio \( c \) as
\[
c := \left( 4 \left\lceil \frac{1}{4c_0} \right\rceil \right)^{-1}.
\]

As we still want to hold on the property that the smaller \( n \) copies fit exactly in the diamond, we also have to adjust \( \alpha \) and set
\[
\alpha := 2 \arctan(2c).
\]

This also implies, that we have to modify our OSC–set \( O \) slightly. We choose the union of the diamond and a circle around \( p \) with radius \( c \) as new set \( O \). The different steps of our construction can also be seen in Figure 12.

Let us now consider the word space. For this, we denote the cells in line from top to bottom by

\[ 1, \ldots, n_0 \]

Figure 12 should help to understand this word space.
By construction it holds that two cells \( w \) with \( b \) as a second case we consider the cells from top to bottom. Those cells are coded by \((B2)\) which implies it holds that this has to be valid for all words around \( p \) and we get, that

\[
m(a_11) = m(w_1) = m(w_2) = m(a_21)
\]

and further \( m(a_1) = m(a_2) \).

This has to be valid for all words around \( p \) and we get, that

\[
m(a) = m(N)
\]

must hold for all \( a \in \{ \frac{n}{2}, \frac{n}{2} + 1, n_0 + 1, \ldots, N \} \).

As a second case we consider the cells from top to bottom. Those cells are coded by

\[
b1 \quad \text{and} \quad bn_0
\]

with \( b \in \{1, \ldots, n_0\} \).

By construction it holds that two cells \( w_1 \) and \( w_2 \) intersect in the way, that

\[
w_1 = b_1d \quad \text{and} \quad w_2 = (b_1 + 1)d
\]

with \( b_1 \in \{1, \ldots, n_0 - 2\} \) and \( d \in \{1, n_0\} \).

Again the first condition of \((B2)\) has to be fulfilled, which implies

\[
m(b_1d) = m(w_1) = m(w_2) = m((b_1 + 1)d)
\]

and thus

\[
m(b_1) = m((b_1 + 1))
\]

must hold. By induction it follows, that

\[
m(b_1) = m(1)
\]
must hold for $b_1 \in \{1, \ldots, n_0 - 1\}$. We are now nearly finished. The last two equivalent words we have to consider are

$$w_1 = (n_0 - 1) n_0 \quad \text{and} \quad w_2 = n_0 1.$$  

By the previous results we have, that

$$m(w_1) = m( (n_0 - 1) n_0 ) = m( (n_0 - 1) ) m(n_0) = m(1) m(n_0)$$

must hold. This implies also that

$$m(w_1) = m(w_2)$$

holds independent of the choice of $m(1)$ and $m(n_0)$ and \lfloor B2 \rfloor is fulfilled. Thus we can choose

$$m(n_0) \in (0, 1) \quad \text{and} \quad m(b_1) = \frac{1 - m(n_0)}{N - 1}$$

for $b_1 \in \{1, \ldots, n_0 - 1\}$ and at the same time \lfloor B2 \rfloor holds. This gives us an example with an inhomogeneous mass distribution and cells with $R(w) = n \in \mathbb{N}$.

The examples of this section show some interesting facts and allow us to understand the condition \lfloor B2 \rfloor in a more deeper way. At the same time those examples demonstrate in a clear way that \lfloor B2 \rfloor is technically a proper condition, but in practice very complex.

6 Analyze \lfloor B2 \rfloor with an algorithm

As we have already seen in section 4, we can create the Sierpiński gasket with several various iterated function systems. Depending on the chosen function system, it varies, how many weights can be chosen and how many weights are fixed. This depends purely on the topology, while the attractor of the IFS stays the same although the distribution of the weights differ.

This fact raises the question how many weights we can choose, if we fix the attractor but vary at the same time the iterated function systems. At the same time we can ask how the number of free weights is distributed. It is clear that this is bounded by $N - 1$ parameters, since the weights have to sum up to 1 and thus the last chosen weight $m(a)$ has to be

$$m(a) = 1 - \sum_{i=1, i \neq a}^{N} m(i).$$

This implies that $N - 1$ free weights are the optimal case and it is by now not clear, that this can be achieved for every attractor of an IFS (for the Sierpiński gasket we already know this).

It seems to be impossible to answer this question in general for all possible fractals (or a wide class of fractals like all p.c.f.-fractals or all nested fractals) as this depends on each topology of every fractal. But what about a single fractal? Can we answer this question for a given fractal positive? For the Sierpiński gasket section 4 answers this question positive, since we found for every case an example. If we take a look at the 3-level Sierpiński gasket $SG_3$ this question seems to be quite harder. We get a lot of equations which have to be fulfilled and we cannot solve them easily. So it seems to be quite hopeless to find easily suitable examples for every number of free weights despite the fact that they may not exist. The big problem is that we would have to check every possible IFS by hand and calculate how many free weights are possible. This may be possible for the Sierpiński gasket where we would have to handle $6^3 = 216$ different iterated function systems, but if we take a look at the $SG_3$, this seems to be a big task since there are already $6^6 = 46 656$ possible iterated function systems. This problem gets even more harder, if we consider other fractals with more possible mappings or with a larger alphabet.

For this reason we want to develop a computer algorithm\footnote{implemented in Python 3.6, see \cite{Pyt}} which calculates us the number of free parameters to all possible iterated function systems with the same attractor. Based on this we are also able to gather how the free parameters are distributed.

The big idea of our algorithm is the following: we fix a particular fractal and put IFS, which
are equivalent under rotating or flipping into an equivalence class. After this, we go through all equivalence classes and pick one representant. For each representant we set up the equations which have to be fulfilled. Since there are typically more equations than parameters, we simplify the equations up to a certain point. We then apply the implicit function theorem and receive as the number of free weights the dimension of the subspace, which we save (e.g. in a file). After this, we continue with the next IFS until we finished.

Our first step is to fix a certain fractal with \( N \) similarities and an alphabet \( A = \{1, \ldots, N\} \). This could be for example the Sierpiński gasket, the 3-level Sierpiński gasket or any other fractal. We have to check by hand, if the equivalence relation is in fact equivalent according to Assumption \( [A] \). Luckily, we only have to check this for one particular realization, since all other realizations are permutations of the word space which won’t effect the property of equivalence. Further we have to check, if the homogeneous case solves the problem. This has it origin in \( \{B1\} \) where we stated that the Martin kernel in the homogeneous case exists. This has also checked only once, since the weights are equal and the equations stay the same under renaming of the variables.

In the next step we have to put all IFS into equivalence classes, since this saves a lot of calculating time. For a better understanding let us start with two small examples.

**Example 6.1.** Let us consider the Sierpiński gasket. We want to take a closer look at the iterated function system from Example 4.2. In this case the top triangle were shrank, the bottom left was also shrank and the bottom right triangle was rotated by \( \frac{2\pi}{3} \) and flipped. The word space (of the second depth) can be represented by the list

\[
[[11, 12, 13], [21, 22, 23], [32, 33, 31]].
\]

and can also be found in Figure 13.

Let us now consider what happens, if we just flip the whole fractal along the y-axis without any rotation. After this, the word space can be represented by

\[
[[11, 13, 12], [32, 33, 31], [21, 23, 22]].
\]

We can rename the labels in such a way, that the top triangle starts with “1”, the bottom left with “2” and the bottom right with “3”. For this, we rename “2” to “3” and “3” to “2”. The label of “1” stays the same. If we do so, we receive

\[
[[11, 12, 13], [23, 22, 21], [31, 32, 33]].
\]
This is also illustrated in Figure 13. As we can see, we receive a different word space. The equations on this word space are however the same. For this reason we only have to find the number of free parameters of the first word space to receive the number of free parameters of the second word space.

For a deeper understanding let us consider another mapping of the whole word space.

**Example 6.2.** Again, we consider the Sierpiński gasket and use the same word space 

\[
[[11, 12, 13], [21, 22, 23], [32, 33, 31]]
\]

as in Example 6.1 as starting word space. This time we rotate the fractal by \( \frac{2}{3}\pi \) and flip the fractal over. We receive 

\[
[[22, 21, 23], [12, 11, 13], [31, 32, 33]].
\]

Let us again rename the word space. We rename “1” to “2”, “2” to ”1” and “3” does not change. After renaming we receive 

\[
[[11, 12, 13], [21, 22, 23], [32, 33, 31]]
\]

and in Figure 14 this is also illustrated. If we compare this with our starting word space, we see that they are identical. Thus this mapping will not generate an other representant in the equivalence class.

Indeed this equivalence class consists in total of three different word spaces. The last missing one is 

\[
[[11, 13, 12], [21, 22, 23], [31, 32, 33]]
\]

which we would receive if we rotate the starting word space by \( \frac{2}{3}\pi \) but without flipping.

From Example 6.2 we then know that all these three different word spaces have exactly one free parameter.

Examples 6.1 and 6.2 give already a glue how we can put all word spaces into equivalence classes. We simply have to apply all possible rotations and flippings onto a starting word space and rename it afterwards. Since it is nearly impossible to predict which word spaces are put into equivalence classes we do the following: we go through all possible IFS and determine their equivalence class. After this, we only consider unique equivalence classes and drop all duplicates. This is also summed up in Listing 1 as a pseudo algorithm.

In fact, we do not have to put the IFS into equivalence classes. But since the calculation of the free parameters takes a lot of computing time, this reduction is very welcome, since we only have to consider between 8-12% of all cases, which depends on the fractal. Further the calculation time for determine all equivalence classes is relatively short.

In a next step we want to go through all equivalence classes of iterated function systems. Since we do the same thing for every iterated function system, we want to fix a particular IFS and keep in mind that we want to iterate over all IFS. So, the next steps will be done for every IFS. We can set up a list of equations, which have to be fulfilled regarding to Proposition 3.3. Every equation is of the form of 

\[
m(v_1)m(v_2)\cdots m(v_n) = m(w_1)m(w_2)\cdots m(w_n)
\]

for equivalent words \( v \sim w \) with \( v = v_1\ldots v_n \) and \( w = w_1\ldots w_n \).

This list of equations has to be replenished by 

\[
\sum_{a=1}^{N} m(a) - 1 = 0.
\]
We can now solve those equations with a computer algebra system. For this we can use for example the Python-package SymPy (see [MSP+17]), which can solve also non-linear systems. Unfortunately SymPy returns on some IFS an error and says, that the list of equations is not solvable by its build-in function nonlinsolve. As a first consequence of this, we want to simplify the equations in parts on our own. For this, we have several possibilities, which we will discuss in the following methods.

**Method 6.3 (“delete double equations”).** If two equations are identical, we can reduce our equation list by one of those equations and consider only the reduced equation list. For example we can reduce
\[ m(a)m(b) = m(c)m(d) \]
\[ m(c)m(d) = m(a)m(b) \]
to
\[ m(a)m(b) = m(c)m(d). \]

This method seems to be quiet trivial, nevertheless one should mention this method. In addition this allows us later to reference to it and we may build up a more complex algorithm using this besides the following methods.

**Method 6.4 (“delete factorial variables”).** We can shorten any factor \( m(i), i \in A \), which occurs on both sides, since \( m(i) > 0 \). For example we can replace
\[ m(a)^k m(b) = m(a)^l m(c) \] with \( k, l \in \mathbb{R} \)
by
\[ m(b) = m(a)^{l-k} m(c). \]

This allows us to shorten unnecessary factors without touching the statement of the equation. This fact is essential, since we only want to simplify our equations for SymPy. The next method will also preserve the statement.

**Method 6.5 (“replace polynomial terms”).** We can extract roots or exponents, if it occurs on one side with the same exponent. This is possible, since \( m(a) > 0 \) for all \( a \in A \) must hold. For example we can replace
\[ m(a)^k m(b) = m(b)m(c)^2 \] with \( k \in \mathbb{R}\{0\} \)
by
\[ m(a) = (m(b)m(c)^2)^{-k} \]

The purpose of this method is the observation that SymPy has several problems with handling exponents, especially with rational exponents. This method seems to be counterproductive, since we modify the equations in the way, that (almost surely) rational exponents occur. The following method exploits another side effect, namely the occurrence of an isolated variable at one side.

**Method 6.6 (“substitute variables”).** We can substitute variables, if they are isolated on one side and the exponent equals 1 (which we can achieve by Method 6.5). For example we can replace
\[ m(a)m(b) = m(c)m(d) \]
\[ m(a) = \sqrt{m(c)m(f)} \]
by
\[ \sqrt{m(c)m(f)}m(b) = m(c)m(d) \]
\[ m(a) = \sqrt{m(c)m(f)}. \]

Further we know that the substituted variable cannot be free and we reduced our problem by one dimension.

All these introduced methods are interesting, but alone they are not really useful. Thus we want to combine them and make them a very powerful weapon to simplify our equations.
Method 6.7 ("simplify equations"). We combine the Methods 6.3 - 6.6 to the following recursive method, called simplify equations.

We start with all equations and call them free equations. Further we start with free variables, which are all variables. For the recursion we implement the fix equations and fix variables as empty.

1. replace in free equations all polynomial terms (by Method 6.5)
2. delete in free equations all factorial variables (by Method 6.4)
3. delete in free equations all double equations (by Method 6.3)
4. if we can substitute a free variable in free equations (by Method 6.6), substitute the variable and save it as a fix variable (and delete it from free variables). Further add the equation defining the variable as fix equation and delete it from free equations. After this, start again at 1. Otherwise we are done.

With this method we receive a list of free equations (potentially empty), a list of fix equations (defining the fix variables), a list of free variables and a list of fix variables.

After we apply Method 6.7 to our equations we receive in most times an empty list of free equations. Further is the list of fix equations and the list of fix variables of the same length, since every equation in fixed equations provides a definition of a fixed variable. If the list of free equations is empty it is clear, that we can choose for every variable in the list of free variables a different value, where maybe some restrictions must hold. Nevertheless we can choose them independently from each other and those variables span up a subspace of $(0, 1)^N$.

Since we are only interested in the dimension of this subspace (where the dimension equals the number of free parameters) we have to consider a mathematical way to verify that those free variables form a proper subspace.

For this, we want to apply the implicit function theorem. To do so, let us first fix some notations.

**Definition 6.8.** Let us fix a specific IFS, its word space and its starting equations. Let us apply Method 6.6 on the starting equations.

For simplicity let us denote the free variables by $x_1, \ldots, x_k$ and the fixed variables by $y_1, \ldots, y_m$.

These variables representing one specific $m(a)$ with $a \in A$ and $k + m = N$ must hold. Further we want to note by $x := (x_1, \ldots, x_k)$ and $y := (y_1, \ldots, y_m)$ those variables combined as a vector.

We can then write the list of all $M$ equations (the combination of free and fixed equations) as

$$g_1(x, y) = h_1(x, y)$$

$$\vdots$$

$$g_M(x, y) = h_M(x, y)$$

where $g_i(x, y)$ and $h_i(x, y)$ are polynomials in $x_1, \ldots, x_k, y_1, \ldots, y_m$.

We can define $F_i(x, y) := g_i(x, y) - h_i(x, y)$ for $i = 1, \ldots, M$ and rewrite equations (5) as a single function by

$$F : (0, 1)^k \times (0, 1)^m \rightarrow \mathbb{R}^M, F(x, y) := \left( \begin{array}{c} F_1(x, y) \\ \vdots \\ F_M(x, y) \end{array} \right)$$

Finally we are interested in solutions with $F(x, y) = 0$

and moreover to maximize $m$ and verify, that $m$ can be chosen maximal. $m$ is then the number of free parameters.

With this notation it is much more clearer, what we want to do. Further this notation implies already an idea, in which direction our further considerations can go.

First of all, let us notice that always $m \leq M$ holds. This comes from the fact, that for every variable $y_i$ an equation exists, namely in fix variables. Thus we can split up $M$ into $M = M_1 + m$, where $M_1$ equals the number of equations in free equations and $m$ the number of equations in fix equations.

If further $M_1 = 0$ and thus $M = m$ holds, we can apply the implicit function theorem.
Proposition 6.9 (Application of the implicit function theorem, cf. [For17]). Consider a specific IFS. Let us denote by $F$ the simplified equations as in Definition 6.8. If $M = m$ holds, we are facing the following problem:

$$F : (0,1)^k \times (0,1)^m \to \mathbb{R}^m, F(x,y) := \begin{pmatrix} F_1(x,y) \\ \vdots \\ F_m(x,y) \end{pmatrix}$$

where $F$ is a continuous and differentiable function.

Let $J$ be the Jacobian matrix defined by

$$J(y) := \frac{\partial F}{\partial y} := \frac{\partial (F_1, \ldots, F_m)}{\partial (y_1, \ldots, y_m)} := \begin{pmatrix} \frac{\partial F_1}{\partial y_1} & \cdots & \frac{\partial F_1}{\partial y_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_m}{\partial y_1} & \cdots & \frac{\partial F_m}{\partial y_m} \end{pmatrix}$$

and define

$$x_{\text{hom}} := \left( \frac{1}{N}, \ldots, \frac{1}{N} \right) \text{ with } k \text{ entries},$$

$$y_{\text{hom}} := \left( \frac{1}{N}, \ldots, \frac{1}{N} \right) \text{ with } m \text{ entries}.$$ 

If $\det J(y_{\text{hom}}) \neq 0$ holds, we can choose $k$ variables arbitrarily in a small neighborhood $U_1 \subset (0,1)^k$ of $x_{\text{hom}}$ and thus the IFS has $k$ free parameters.

Proof. The proof is in fact an application of the implicit function theorem. Recall that if the Jacobian $J$ is invertible at the point $y_{\text{hom}}$, then there exists an open neighborhood $U_1 \subset (0,1)^k$ of $x_{\text{hom}}$, an open neighborhood $U_2 \subset (0,1)^m$ of $y_{\text{hom}}$ and a continuous differentiable function $G : U_1 \to U_2$ with $G(x_{\text{hom}}) = y_{\text{hom}}$ such that

$$F(x,G(x)) = 0 \quad \text{holds for all } x \in U_1.$$ 

The exact form of $G$ is for us irrelevant, but we can choose in $U_1$ each coordinate of $x$ independently from each other, thus we can choose $k$ weights and the IFS has $k$ free parameters.

Remark 6.10. We can apply Proposition 6.9 even if in the case of $m < M$. For this, we have to extend the fixed variables by $m_0 = M - m$ free variables. We have to be careful with the choice, since we could choose variables which lead to a non-invertible Jacobian matrix. Thus we have to consider in this case all possible subsets of free variables with cardinality $m_0$. We set up for each the Jacobian and if $\det J(y_{\text{hom}}) \neq 0$ we have found a constellation such that the implicit function theorem is applicable. It then follows, that the IFS has $N - M$ free parameters.

In total we now know how to determine the number of free parameters. One thing still remains, which is the relatively high computing time.

We can speed up our algorithm if we split up our problem into multiple parts. To be precise, we split this up in the number of kernels of the computer and each kernel receives a part. Thus we can calculate parallel. A further acceleration would be possible, if we split up our algorithm to multiple computers, but we skip this here since it would make our algorithm harder to understand and the essential part of splitting up is already contained in the parallel computing–part.

All together we can set up a pseudo algorithm, which can be found in Listing 2. In line 15 and 23 there is a statement for raising a Warning. In fact, up to now this has not occurred on any considered fractal. If a warning would raise we should extend our Method 6.6.

This algorithm has some nice properties, but at the same time also some disadvantages. One of this disadvantages is the total time our algorithm needs for his total computation. This comes from the fact that we check all equivalence classes of iterated function systems. For this, recall that the number of all IFS grows very fast by the law

$$\#\text{IFS} = |\text{different mappings for a single copy}|^N.$$
choose the fractal to consider
all_IFS := list of all possible IFS
all_eq_classes := determine all equivalence classes of all_IFS (compare to listing 1)

for each kernel:
  pick an element from all_eq_classes which was until now not considered:
  set up list of equations associated to the IFS
  apply Method 6.7 and receive free_equations, fix_equations, free_variable,
  fix_variables (with the notation of Definition 6.8)
  if length(free_equations) = 0:
    set up Jacobian matrix J
    if det(J(y_hom)) ≠ 0:
      save M as number of free parameters
    else:
      raise Warning
  else:
    for all combinations of subsets of free_variable:
      combine combination and fix_variable
      set up Jacobian matrix J
      if det(J(y_hom)) ≠ 0:
        save M as number of free parameters
      else:
        raise Warning
merge results of the different kernels
return list with IFS, number of free parameters and length of its equivalence class

Listing 2: Pseudo algorithm for calculating the number of free parameters to all IFS generating a specific fractal.

The number of equivalence classes cannot calculated in an explicit form, but is approximately

#equivalence classes ≈ |different mappings for a single copy|^{N-1}.

Our algorithm needs more calculating time if the number of equivalence classes raises. Since the amount grows exponentially, this grows more worse, if either the alphabet gets bigger or if we have a higher number of possible small mappings. The fact, that we use parallel computing can only compensate this exponential growing rate in a minor way.

Further it should not be underestimated that the number of equations raises also the computing time. Thus the time to terminate the algorithm depends in the first place on the number of IFS and in the second place on the numbers of equations/variables (and of course on the used hardware).

Let us now apply our algorithm to several (common) fractals. Of course, we want to investigate the already introduced fractals Sierpiński gasket and the 3-level Sierpiński gasket. Further we want to consider the Vicsek fractal (see [Vic92]), the Pentagasket with and without hole (see [ASST03, Ima00]) and the Hexagasket (see [Str06]). The Hexagasket could also be considered with a filled center, but for now we skip this, since it would exceed the computing time, since there are about 35.8mio different IFS and approximately 3mio equivalence classes.

As a small reminder contains Figure 15 (resp. the subfigures) the associated attractors of these fractals.

After we execute our algorithm we receive for every fractal a characterization of the associated space of solutions. In Table 1 the results are summarized.

First of all the length of the alphabet N is listed, which should be only a small reminder. Further the table contains the number of different small mappings for each small copy and the number of equations which have to be fulfilled such that (B2) holds. Those equations differ for each IFS but the number stays the same since only the words interchange.

The next entry of the table contains the number of equivalence classes which occur. The amount of
Figure 15: The attractors of the Sierpiński gasket, the 3-level Sierpiński gasket, the Vicsek fractal, the Pentagasket (with and without hole) and the Hexagasket.

| N       | SG2 | SG3 | Vicsek | Pentagasket | Pentagasket without hole | Hexagasket |
|---------|-----|-----|--------|-------------|--------------------------|------------|
| number of different mappings | 6   | 6   | 8      | 10          | 10                       | 12         |
| number of equations which have to be fulfilled | 3   | 9   | 4      | 5           | 25                       | 6          |
| number of equivalence classes | 44  | 7860 | 4360 | 10104       | 100220                   | 250010     |
| 0 free parameter | 194 | 46257 | 20544 | 88025       | 999995                   | 2599398    |
| 1 free parameter | 21  | 399  | 10112 | 11875       | 5                        | 361007     |
| 2 free parameters | 1   | 0    | 2048  | 100         | 0                        | 24075      |
| 3 free parameters | -   | 0    | 64    | 0           | 0                        | 1452       |
| 4 free parameters | -   | 0    | 0     | 0           | 0                        | 51         |
| 5 free parameters | -   | 0    | -     | -           | 0                        | 1          |
| total number of IFS | 216 | 46656 | 32768 | 100000      | 1000000                  | 2985984    |

Table 1: Summarized results about the number of free parameters on different fractals.
equivalence classes is slightly more than the number of different mappings to the power of \((N - 1)\), since every mapping occurs at most once. But since there are some IFS that are invariant under mapping, there are some more.

The next few entries describe the occurrence of free parameters in total numbers. In other words, if we add all up, we receive the total number of IFS, which is as a reminder added at the bottom of the table.

As we can see, there is only one realization of the Sierpiński gasket with two free parameters which is exactly Example 4.3. Further occur only 21 IFS with one free parameter and the majority of cases have zero free parameters.

The 3-level Sierpiński gasket is generated by six small copies and thus five free parameters would be the maximum of possible free parameters. Indeed this does not occur. Neither four, three or two parameters do. As we can see, there are only 399 IFS (which is about 0.9\%) with one free parameters and thus this is very uncommon. The remaining IFS have all zero free parameters which implies, that those IFS only fulfill (B2) in the homogeneous case. One of the reasons for this fact is that there are in total nine equations which have to be fulfilled, making it harder to find any solution besides the homogeneous case.

On the Vicsek fractal the number of free parameters are wider distributed, for example there are 64 IFS with three free parameters and over 12,160 IFS with one or two free parameters. Nevertheless an IFS where the full potential of four free parameters can be chosen is missing. The Pentagasket behaves in a total other way than the Vicsek fractal, even if it has also an alphabet of length \(N = 5\). Most IFS have zero free parameters, about 11.8\% have one free parameter and only 100 have two free parameters. Those 100 IFS are contained in 14 different equivalence classes.

If we add further a small copy in the middle, we receive the Pentagasket without hole. It is obvious, that this restricts the number of free parameters. Indeed this restricts the choice in a massive way. There is only one (!) equivalence class containing five IFS, where we can choose one free parameter. On all other IFS we are not able to choose any free parameter.

Last but not least we want to take a look at the Hexagasket. The distribution of free parameters is again relatively broad and we are indeed able to choose an IFS with the maximum of five free parameters. On the other hand we have again on most IFS choose only zero free parameters. This schema occurs on all fractals and we can discover another schema: if there are more equations which have to be fulfilled it is harder to choose many free parameters.

The calculation time on the computer differs from fractal to fractal in a great manner. The following computing times refer all to the same computer with 60 kernels. The Sierpiński gasket took about only a few seconds to compute all 44 equivalence classes. The 3-level Sierpiński gasket \(SG_3\) took about 1,700 seconds, whereas the Vicsek only took 146 seconds and the Pentagasket took 563 seconds. Compared to the number of equivalence classes performs the \(SG_3\) thus relatively slow.

The computing time of the Pentagasket without hole took extremely long and our computer needed about 191,150 seconds (or approx. 2d 5h). The Hexagasket was again calculated relatively fast and took only 26,400 seconds respectively 7h 20min.

In future we could use this algorithm to investigate also other fractals. For this we only need to code the needed equations and the definition of the small copies, which can be done relatively quickly. Also the determination of the equivalence classes can be done quickly. The most restricting part is indeed the calculation of the free parameters. The Pentagasket without hole indicates that the number of equations is the main factor for a long calculating time, which will be the limiting factor of further considerations.

References

[ASST03] B. Adams, S. A. Smith, R. S. Strichartz, and A. Teplyaev. The spectrum of the laplacian on the pentagasket. In Peter Grabner and Wolfgang Woess, editors, Fractals in Graz 2001, pages 1–24, Basel, 2003. Birkhäuser Basel.

[DS01] M. Denker and H. Sato. Sierpiński Gasket as a Martin Boundary I: Martin Kernels; Dedicated to Professor Masatoshi Fukushima on the occasion of his 60th birthday. Potential Analysis, 14(3):211–232, May 2001.
[DS02] M. Denker and H. Sato. Reflections on Harmonic Analysis of the Sierpiński Gasket. *Mathematische Nachrichten*, 241(1):32–55, 2002.

[Fal90] K. Falconer. *Fractal geometry: mathematical foundations and applications*. Wiley, Chichester [u.a.], 1990.

[FK19] U. Freiberg and S. Kohl. Martin boundary theory on inhomogenous fractals. *arXiv e-prints*, page arXiv:1907.07499, Jul 2019. Available at https://arxiv.org/pdf/1907.07499.

[For17] O. Forster. *Differentialrechnung im Rn, gewöhnliche Differentialgleichungen*, volume 2 of *Analysis ; 2*. Springer Spektrum, Wiesbaden ; [Heidelberg], 11., extended edition edition, 2017.

[Ham00] B. M. Hambly. Heat kernels and spectral asymptotics for some random sierpinski gaskets. In Christoph Bandt, Siegfried Graf, and Martina Zähle, editors, *Fractal Geometry and Stochastics II*, pages 239–267, Basel, 2000. Birkhäuser Basel.

[Hut81] J. Hutchinson. Fractals and self similarity. *Indiana University Mathematics Journal*, 30(5):713–747, 1981.

[Ima00] A. Imai. *Pentakun, the mod 5 Markov chain and a Martin boundary*. 2000.

[JLW12] H. Ju, K.-S. Lau, and X.-Y. Wang. Post-critically Finite Fractal and Martin boundary. *Transactions of the American Mathematical Society*, 364(1):103–118, 2012.

[Kig01] J. Kigami. *Analysis on fractals*, volume 143. Cambridge University Press, 2001.

[Koc04] Helge von Koch. Sur une courbe continue sans tangente, obtenue par une construction géométrique élémentaire. *Ark. Mat. Astron. Fys.*, 1:681–702, 1904.

[LN12] K.-S. Lau and S.-M. Ngai. Martin boundary and exit space on the Sierpinski gasket. *Science China Mathematics*, 55(3):475–494, Mar 2012.

[LW15] K.-S. Lau and X.-Y. Wang. Denker–Sato type Markov chains on self-similar sets. *Mathematische Zeitschrift*, 280(1):401–420, Jun 2015.

[MSP+17] A. Meurer, C. P. Smith, M. Paprocki, O. Čertík, S. B. Kirpichev, M. Rocklin, A. Kumar, S. Ivanov, J. K. Moore, S. Singh, T. Rathnayake, S. Vig, B. E. Granger, R. P. Muller, F. Bonazzi, H. Gupta, S. Vats, F. Johansson, F. Pedregosa, M. J. Curry, Andy R. T., S. Růžička, A. Saboo, I. Fernando, S. Kulal, R. Cimrman, and A. Scopatz. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103, January 2017.

[Pyt] Python Language Reference, version 3.5. *Python Software Foundation*. Available at https://www.python.org/ documentation at https://docs.python.org/3.5/.

[Str06] R. Strichartz. *Differential equations on fractals. A tutorial*. Princeton University Press, 01 2006.

[Vic92] T. Vécsei. *Fractal Growth Phenomena*. WORLD SCIENTIFIC, 2nd edition, 1992.