A computational approach to finding the Hausdorff dimension of continued fraction Cantor sets

Rowan Woodell
Supervised by: Charles Walkden

September 2023

A dissertation submitted to the University of Manchester
for the degree of Master of Science in the
Faculty of Science and Engineering.

Contents

Abstract

In 2001, Jenkinson & Pollicott computed to 25 decimal places the dimension of the set E_2 of numbers whose continued fraction expansion contains only the numbers 1 and 2, as well as outlining a fast-convergent algorithm for computing the dimension of E_2 and similary sets [JP01]. This dissertation, focusing instead on the entirely analogous set $E_{2,5}$, aims to present their results in a more digestible form, giving relevant insight into background material and illuminating parts of their proofs which may be particularly difficult for those new to the subject area. We also give details of a Python implementation of their algorithm, and examine the theoretical complexity and practical execution time of the algorithm.

Acknowledgements

I would like to thank my supervisor, Charles Walkden, for his patience and invaluable support over the last few months. I am also indebted to my fellow students John and Alex, for staring with me at blackboards and code for more hours than I can count.

1 Introduction

1.1 Overview

The theory of continued fractions allows us to construct fractal sets in a very straightforward way, by taking sets of numbers whose continued fraction expansion contains only numbers from a given subset of $\mathbb{N}$. Despite their simple construction, these sets are complex and difficult to reason about. As we may expect, the study of these sets has applications in number theory – specifically the Markov and Lagrange spectra, integral to the theory of Diophantine approximations [JP01] – as well as further afield, in the study of Dirichlet series [Cus77]. In the journey towards understanding these sets, we take a winding route through many areas of mathematics – number theory, fractal geometry, dynamical systems, ergodic theory, complex analysis, functional analysis, and complexity theory – and tie many disparate results together in order to to form a complete picture of how we can think about these sets, and how we are able to calculate their dimensions.

For the purposes of this dissertation, we are interested in the Hausdorff dimension of the set $E_{2,5}$ of numbers in the interval (0, 1) whose continued fraction expansion consists only of the numbers 2 and 5, though we may equally have chosen any finite set of natural numbers. We follow Oliver Jenkinson and Mark Pollicott's 2001 paper on computing the dimension of $E_{1,2}$ [JP01], providing essential prerequisite knowledge to understanding their work (section 2), before presenting the proofs of their results, adapting them to fit our set of interest $E_{2,5}$ and elaborating on certain less-obvious parts of their proofs (section 3). We then present details and results of our own implementation of their computational method for finding the dimension of $E_{2,5}$ and similarly-defined sets (section 4).

The remainder of section 1 introduces the set $E_{2,5}$ and its construction, as well as a history of prior work on determining the dimensions of these sets (along with other, closely related problems).

The aim of section 2 is to fully acquaint the reader with all necessary background material before delving into the technical proofs. We give a brief recap of fractal geometry, recalling the notion of Hausdorff dimension and relevant results. We then cover some standard results of complex analysis, namely the Estimation Lemma (2.2.8) and Cauchy's Integral Formula (Theorem 2.2.13), which are invoked at key points of the proofs that follow. A brief overview of the general theory of Banach spaces and bounded linear operators follows, which we attempt to elucidate using familiar examples (i.e. square matrices as operators on Euclidean space), before introducing the family of Banach spaces in which we are particularly interested. We then cover Grothendieck's theory of nuclear and trace-class operators on Banach spaces, allowing us to define a trace for general bounded linear operators. Finally, we introduce the transfer operator (also known as the Ruelle-Perron-Frobenius operator) of an iterated map and provide some relevant motivating examples.

Section 3 states the main results of [JP01], and details the proofs. We focus particularly on demystifying the more obtuse parts of the proofs given in the original paper, notably the derivations of the trace formula (p. 26–28) and the power series coefficients (p. 28–28), as well as a full big-O analysis of the convergence rate of estimations of the dimension (p. 30–31).

We conclude in section 4 with details of a Python implementation of the algorithm described in [JP01]. We also compare our computational results to those of Jenkinson & Pollicott and, finally, give an analysis of the time and space complexity of the algorithm and of the performance of our specific implementation.

1.2 The Problem

Given a (possibly infinite) sequence of positive integers $(a_1,\, a_2,\, a_3, \cdots)$, we define the *continued fraction* with coefficients $a_1,\, a_2,\, a_3, \cdots$ by

$$[a_1,\, a_2,\, a_3, \cdots] := \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \cdots}}}.$$

Any such continued fraction uniquely defines a number in the interval $(0, 1)$ [Khi97]. The notation $[a_0;\, a_1,\, a_2, \ldots]$ is often used to mean $a_0 + [a_1,\, a_2, \ldots]$, however in our work we are only interested in numbers between 0 and 1 (i.e. continued fractions with $a_0 = 0$), so we omit this term going forwards. If the sequence $(a_i)_{i\in\mathbb{N}}$ repeats after a finite number of terms, then we denote the repeating part with an overline, i.e.

$$[a_1, \cdots a_k,\, \overline{a_{k+1}, \ldots, a_{k+n}}] := [a_1, \ldots a_k,\, a_{k+1}, \ldots, a_{k+n},\, a_{k+1}, \ldots, a_{k+n}, \ldots].$$

If the sequence has no entries before the repeating part, then we simply write

$$[\overline{a_1, \ldots, a_n}] := [a_1, \ldots, a_n,\, a_1, \ldots, a_n,\, a_1, \ldots].$$

A continued fraction with a repeating part is called *periodic*; if it has no non-repeating part, then it is called *purely periodic.*

Consider the set $E_{2,5}$ of numbers for which all the coefficients a_n in their (infinite) continued fraction expansion are equal to either 2 or 5. The set $E_{2,5}$ is a fractal – it is a non-linear Cantor set with zero one-dimensional Lebesgue measure. We are interested in the dimension of $E_{2,5}$, and of other fractals defined by restricting the coefficients of continued fractions. In particular, we define E_X, with X a finite set of natural numbers, by

$$E_X := \{[a_1,\, a_2, \ldots] \mid a_i \in X\ \forall i\},$$

and we aim to find the dimension of such sets. We discuss the history of these problems – as well as other problems of restricted partial quotients – in section 1.3

The following alternate construction of $E_{2,5}$ provides important context for later work: take the maps $S_2, S_5 : [0, 1] \to [0, 1]$ given by

$$S_i(x) = \frac{1}{i + x}.$$

We may think of these maps as 'inserting' a digit into the first position of the continued fraction expansion of a number. For example, $S_2(x) = \frac{1}{2+x} = [2,\, \frac{1}{x}]$. Now, consider what would happen if we composed infinitely many maps of this form; given a sequence $\underline{i}$ with entries in $\{2, 5\}$, denote $S_{\underline{i}}(x) = (S_{i_1} \circ S_{i_2} \circ \cdots)(x)$. Then regardless of the value of x, $S_{\underline{i}}(x) = [i_1,\, i_2, \ldots]$. The union of the images of all such maps is clearly $E_{2,5}$.

We formalise the above by defining

$$E = \bigcap_{k=0}^{\infty} S_2([0, 1]) \cup S_5([0, 1]),$$

and noticing that $E = E_{2,5}$. Those familiar with fractal geometry and dynamical systems will know that the set E is the *attractor* of the *iterated function system* (IFS) given by $\{S_2, S_5\}$. A related alternative construction, in terms of cookie-cutters, is given in section 2.1.

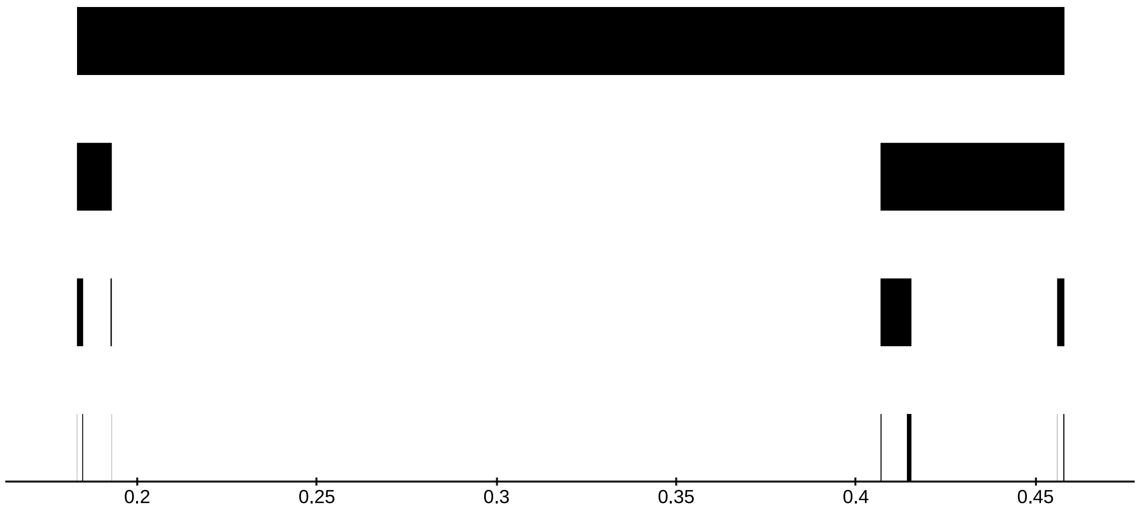

Figure 1: The first four steps in the construction of $E_{2,5}$ via an IFS.

1.3 Related Work

Very broadly, we are interested in finding the dimensions of sets of continued fractions whose partial quotients are restricted in some way, sometimes referred to as problems of *restricted partial quotients*. Various authors have considered problems in this area, some closely related to ours (most notably that of finding the dimension of $E_{1,2}$, more commonly written E_2), and some further afield.

Seemingly the first published exploration of the dimension theory of sets of restricted partial quotients is Vojtěch Jarnìk's 1928 paper concerning Diophantine approximations [Jar28], in which he considers the set E of continued fractions whose partial quotients are bounded. That is,

$$E = \{[a_0;\, a_1,\, a_2, \ldots] \mid \exists\, M \in \mathbb{N} \text{ s.t. } a_i < M\, \forall\, i\}.$$

Jarnìk showed that $\dim E = 1$. Note that the set E is closely related to the particular sets in which we interested in that it is the union of all such sets; any finite subset X of positive integers is necessarily bounded, so the union of all sets $E_X = \{[a_0;\, a_1,\, a_2, \ldots] \mid a_i \in X\}$ is clearly contained in the set E. Conversely, given a point x in E, there exists some M such that for each a_i in the continued fraction expansion of x, $a_i < M$. Hence we can take $X = \{1,\, 2, \ldots, M-1\}$, so in particular X is finite and $x \in E_X$ (note that this X is not necessarily minimal). Therefore the union of all the E_X is precisely E. Jarnìk formulates this equivalently as $E = E_2 \cup E_3 \cup E_4 \cup \cdots$, where E_α is the set of continued fractions whose partial quotients are at most α (this notation has persisted, and appears to be the reason most authors write E_2 instead of $E_{1,2}$). Jarnìk also showed $\dim E_2 > \frac{1}{4}$, and that for all $\alpha > 8$,

$$1 - \frac{4}{\alpha \log 2} \leq \dim E_\alpha \leq 1 - \frac{1}{8\alpha \log \alpha}.$$

By taking the limit $\alpha \to \infty$, we recover the result $\dim E = 1$.

The approach taken in Jarnìk's paper is a relatively straightforward application of measure theory to these sets which are defined number-theoretically. This approach was expanded upon in Irving

Good's 1941 paper *The Fractional Dimensional Theory of Continued Fractions* [Good41]. Good calculated the following bounds for the dimension of E_2:

$$0.5306 < \dim E_2 < 0.5320.$$

Good also considered some other problems of restricted partial quotients, namely those where the sequences of partial quotients are unbounded. In particular, he showed that the set of continued fractions whose partial quotients tend to infinity has dimension $\frac{1}{2}$.

Thomas Cusick found in 1977 [Cus77] that the dimension of E_2 (and, more generally, sets of the form E_X) arose in the study of Dirichlet series. In 1982, further progress was made by Richard Bumby [Bum85]. He used the theory of transfer operators (see section 2.5) to expand on Cusick's work, showing that

$$0.5312 \leq \dim E_2 \leq 0.5314.$$

The use of transfer operators has since become ubiquitous in the study of fractal dimensions; Bumby was among the first to apply the theory to problems of restricted partial quotients. It was also around this time that computational techniques began to be used to find numerical results such as these. The results that follow were all achieved with the aid of computers.

Doug Hensley was able to improve the bound on $\dim E_2$, again using a family of transfer operators. In 1988 [Hen88], he showed that

$$0.531\,280\,49 \leq \dim E_2 \leq 0.531\,280\,51.$$

He improved this estimate in 1994 [Hen96], showing that

$$\dim E_2 \approx 0.531\,280\,506\,277\,205\,141\,6,$$

though he was unable to rigorously justify the decimal digits past the first 6, which were known from his previous bound. Following the work of Konstantin Babenko and S. P. Yur'ev [Bab78, BY78], Hensley restricted the transfer operator used by Bumby to a certain Banach space of complex analytic (or holomorphic) functions – see sections 2.2 & 2.3.

Oliver Jenkinson and Mark Pollicott's 2001 paper [JP01], on which much of our work is based, presented the estimate

$$\dim E_2 = 0.531\,280\,506\,277\,205\,141\,624\,468\,6,$$

correct to the 25 decimal places given. However, these digits were not rigorously justified until their 2018 paper [JP18]. A transfer operator method is used in both papers, with the first working over a Banach space of complex analytic functions, while the second extends this to a similar Hilbert space. This allows for tighter bounds on the error in the estimates, and proves that the estimate given in the first paper is correct to the given decimal places.

In 2017, Richard Falk and Roger Nussbaum [FN17] showed that

$$0.531\,280\,505\,981\,42 \leq \dim E_2 \leq 0.531\,280\,506\,320\,77,$$

which provided a rigorous justification for the first 8 decimal places. In contrast to previous work, they applied the transfer operator theory to a Banach space of real-valued C^k functions for $k \geq 2$.

A more straightforward method, due to Falconer [Fal03], uses the fact that $\dim E_2$ must be strictly between the dimensions of the attractors of two linear IFSs. The dimension of the attractor of a linear IFS is straightforward to calculate; it is the unique s such that

$$\sum_i c_i^s = 1,$$

where the c_i are the similarity ratios of the maps in the IFS. This method gives the bound $0.44 < \dim_H E_2 < 0.66$. Below, we apply Falconer's method to find a bound for $\dim E_{2,5}$.

We take the maps $S_2,\ S_5 : D \to D$, with $D = \left[[\overline{2;\ 5}]\ ,\ [\overline{5;\ 2}]\right] = [\frac{1}{5}(5+\sqrt{35}),\ \frac{1}{2}(5+\sqrt{35})]$, and consider the IFS $\{S_2,\ S_5\}$, with

$$S_2(x) = 2 + \frac{1}{x},\ \ S_5(x) = 5 + \frac{1}{x}.$$

Note that the attractor F of this IFS consists of two copies of $E_{2,5}$, so we have $\dim_H E_{2,5} = \dim_H F$. Also notice that D is chosen to ensure that $S_2(D),\ S_5(D) \subseteq D$; indeed

$$\begin{aligned} S_2(D) &= \left[[\overline{2;\ 5}]\ ,[2;\ \overline{2,\ 5}]\right] \subseteq D, \\ S_5(D) &= \left[[5;\ \overline{5,\ 2}]\ ,[\overline{5;\ 2}]\right] \subseteq D. \end{aligned}$$

Applying the Mean Value Theorem gives that for $x,\ y \in D$ with $x \neq y$,

$$\frac{S_i(x) - S_i(y)}{x - y} = S_i'(z_i),$$

for some $z_i \in D$. Hence, for $i = 2,\ 5$, we have

$$\inf_{z \in D} |S_i'(z)| \leq \frac{|S_i(x) - S_i(y)|}{|x - y|} \leq \sup_{z \in D} |S_i'(z)|.$$

We have $S_2'(z) = S_5'(z) = \frac{-1}{z^2}$, so it follows that

$$(\tfrac{1}{2}(5+\sqrt{35}))^{-2} \leq \frac{|S_i(x) - S_i(y)|}{|x - y|} \leq (\tfrac{1}{5}(5+\sqrt{35}))^{-2}.$$

For a linear IFS, we have that $\dim_H F = s$, where s is the unique solution to $\sum_{i=1}^n c_i^s = 1$. In that case, we have $|S_i(x) - S_i(y)| = c_i|x - y|$ for all $x,\ y \in D$, whereas here we have an inequality of the form $d_i|x-y| \leq |S_i(x) - S_i(y)| \leq c_i|x-y|$. Intuitively, we may assume that $t \leq \dim_H F \leq s$, where $t,\ s$ are the solutions to $\sum_{i=1}^n d_i^t = 1$, $\sum_{i=1}^n c_i^s$. In fact, this assumption holds under certain (very general) conditions. The upper bound is immediate from the result for the dimension of a linear IFS, while the lower bound requires a short argument involving the Mass Distribution Principle; see [Fal03], p. 135-6.

Computing t and s for our example gives that $0.204 < \dim E_{2,5} < 0.444$.

Falconer also notes that by instead considering the four maps given by

$$(S_i \circ S_j)(x) = i + \frac{1}{j + \frac{1}{x}} \qquad (i,\ j = 1,\ 2),$$

one can find better bounds on the dimension of E_2. For $E_{2,5}$, we note that

$$(S_i \circ S_j)'(x) = \frac{1}{jx+1} - \frac{jx}{(jx+1)^2}$$

for $i,\, j \in \{2,\, 5\}$, which allows us to find the infima and suprema as before. Then a similar argument to the above gives that $0.2419 \leq \dim E_{2,5} \leq 0.3424$.

Further, by repeatedly composing maps we can achieve reasonable estimates for the dimension of the attractor of any non-linear IFS. This may seem a promising approach, but – for an IFS with n maps – each successive estimate involves finding n times as many derivatives as the previous one. So this method is exponential in the number of compositions, with no guarantee on the convergence rate; indeed, with two iterations we can't be confident in even a single decimal digit. In section 3, we show that the method outlined in [JP01] converges super-exponentially.

2 Background

2.1 Recap of Fractal Geometry

We give a brief overview of some prerequisite concepts from fractal geometry. We follow the relevant main results from a project preceding this dissertation, which in turn follows Falconer's work in [Fal03, Fal97].

Given a set $F \subseteq \mathbb{R}^n$, recall that the *diameter* of F is the greatest distance between any pair of points in F. Also recall that a δ-cover of F is a cover of F consisting of sets of diameter at most δ. Then we have the following definition:

Definition 2.1.1. The *s-dimensional Hausdorff measure* of a set F is given by

$$\mathcal{H}^s(F) = \lim_{\delta \to 0} \mathcal{H}^s_\delta,$$

where

$$\mathcal{H}^s_\delta = \inf \left\{ \sum_{i=1}^{\infty} \operatorname{diam}(U_i)^s \mid \{U_i\} \text{ is a } \delta\text{-cover of } F \right\}.$$

This provides us with our notion of dimension.

Definition 2.1.2. The *Hausdorff dimension* of a set F is given by

$$\dim_H F = \inf\{s \geq 0 \mid \mathcal{H}^s(F) = 0\}.$$

When it is clear that we mean the Hausdorff dimension, we refer simply to the *dimension* of a set.

Some fractal sets may be defined as the invariant sets of certain classes of dynamical systems. The most intuitive perspective is that of iterated function systems (IFSs), but this approach presents us with limitations when the maps are non-linear. We instead consider cookie-cutter functions, which can be thought of as the inverse of a corresponding IFS.

Definition 2.1.3. Let D be the union of finitely-many disjoint subintervals of $[0, 1]$, and let $T : D \to [0, 1]$ be a function such that:

(i) T is bijective on each subinterval that comprises D;

(ii) T' is continuous;

(iii) $|T'(x)| > 1$ for all $x \in D$.

Then we call T a *cookie-cutter function*, and define the *repeller* of T by

$$E = \{x \in [0, 1] \mid T^k(x) \text{ is defined and in } D \text{ for all } k = 0, 1, 2, \ldots\}.$$

We call E a *cookie-cutter set.*

Example 2.1.4. Define $T : (0, 1) \to [0, 1)$ by $Tx = \frac{1}{x} \mod 1$, i.e. T maps a number between 0 and 1 to the decimal part of its reciprocal. The map T is called the *Gauss map*. We take $D = (\frac{1}{3}, \frac{1}{2}] \cup (\frac{1}{5}, \frac{1}{6}]$ and restrict T to D. Now T is indeed bijective on $(\frac{1}{3}, \frac{1}{2}]$ and $(\frac{1}{5}, \frac{1}{6}]$, and we have $T'(x) = -\frac{1}{x^2}$. Since $0 < x < 1$, we indeed have $|T'(x)| > 1$ for all $x \in D$. Hence T is a cookie-cutter, and we claim that the repeller of T is $E_{2,5}$. To see this, we write

$$Tx = \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor = \left\{ \frac{1}{x} - i \text{ if } x \in \left(\frac{1}{i+1}, \frac{1}{i} \right] \right. .$$

We then consider the inverse branches of T: set $T_i(x) = \frac{1}{x} - i$, then $T_i^{-1}(x) = \frac{1}{i+x}$. Recalling work on the IFS in section 1.2, we can see that $T_i^{-1} = S_i$. The correspondence between IFSs and cookie-cutters (see [Fal97]) tells us that the repeller of T is precisely the attractor of the IFS given by $\{T_2^{-1}, T_5^{-1}\}$, which we know to be $E_{2,5}$.

The repeller of a cookie-cutter is generally a fractal, and we can use the following concept from the thermodynamic formalism to write a formula for the dimension of a cookie-cutter set.

Definition 2.1.5. Let ϕ be a Lipschitz function, T a cookie-cutter. Then the *(topological) pressure* of ϕ (with respect to T) is given by

$$P(\phi) = \lim_{n\to\infty} \frac{1}{n} \log \sum_{x \mid T^n x = x} \exp\left(\phi(x) + \phi(T(x)) + \cdots + \phi(T^{n-1}(x))\right).$$

We have the following result (see [Fal97] p. 77).

Theorem 2.1.6 (The Dimension Formula). *Let T be a cookie-cutter with repeller E, and let s be the unique real number satisfying*

$$P(-s \log |T'|) = 0.$$

Then $\dim_H E = s$ *and* $0 < \mathcal{H}^s(E) < \infty$.

This is a very neat result, but – as discussed in the prior project – it is not reasonable to calculate in practice due to its slow rate of convergence. This is precisely the issue addressed in [JP01], for the sets E_X, where X is a finite set of natural numbers.

2.2 Complex Analysis

We require some standard results from complex analysis, with which some readers may not be familiar. A brief overview follows; see [Ahl79] for a more complete treatment.

Definition 2.2.1. A complex function f is *holomorphic* (sometimes *complex analytic*, or just *analytic*) on an open set U if it is complex differentiable at every point in U.

Definition 2.2.2. A *path* is a continuous function $\gamma : [a,b] \to \mathbb{C}$, where $[a,b]$ is a real interval. A path is *smooth* if γ is differentiable (in the one-sided sense at a and b), with γ' continuous. The *length* of a path γ is defined by

$$\text{length}(\gamma) = \int_a^b |\gamma'(t)|\, dt.$$

Definition 2.2.3. A *contour* γ is a collection of smooth paths $\gamma_1, \ldots, \gamma_n$ such that the end-point of γ_r coincides with the start-point of γ_{r+1} for $1 \le r \le n-1$. We write

$$\gamma = \gamma_1 + \cdots + \gamma_n.$$

If the end-point of γ_n coincides with the start-point of γ_1, we call γ a *closed contour*. If each of the γ_r is a straight line-segment, then we call γ a *polygonal arc*. We define $\text{length}(\gamma) = \text{length}(\gamma_1) + \cdots + \text{length}(\gamma_n)$.

Definition 2.2.4. An open set $D \in \mathbb{C}$ is called a *domain* if every pair of points in D can be connected by a polygonal arc in D.

Example 2.2.5. In particular, the path given by $\gamma(t) = re^{it}$ with $0 \le t \le 2\pi$ is a closed path – it is a circle of radius r centred at 0. The interior of this circle is the open disk D with radius r centred at 0. D is clearly a domain.

For the remainder of this section, D denotes a domain.

Definition 2.2.6. Let $f : D \to \mathbb{C}$ be a complex function and $\gamma : [a, b] \to D$ a smooth path. Then define

$$\int_\gamma f = \int_\gamma f(z)\,dz = \int_a^b f(\gamma(t))\gamma'(t)\,dt.$$

If $\gamma = \gamma_1 + \cdots + \gamma_n$ is a contour in D, we define

$$\int_\gamma f = \int_{\gamma_1} f + \cdots + \int_{\gamma_n} f.$$

Remark 2.2.7. Note that the integral over a path is independent of the choice of parameterisation of the path.

Lemma 2.2.8 (Estimation Lemma). *Let $f : D \to \mathbb{C}$ be continuous, and γ a contour in D. Suppose $|f(z)| \le M$ for all z on γ. Then*

$$\left|\int_\gamma f\right| \le M \cdot \text{length}(\gamma).$$

Proof. Omitted. □

This result is used in section 3.2 to bound the norm of a linear functional, which allows us to find a suitable bound on the eigenvalues of a linear operator.

Definition 2.2.9. Let γ be a closed contour in $\mathbb{C}$, $z \in \mathbb{C}$ not on γ. Then the *winding number* of γ around z, denoted $w(\gamma, z_0)$, is the number of times γ "winds" around z, counting positively for anticlockwise turns and negatively for clockwise turns. The winding number must be an integer.

Example 2.2.10. Let γ be as in Example 2.2.5, then $w(\gamma, 0) = 1$. If we instead parameterised the same path by $\beta(t) = re^{-it}$ with $0 \le t \le 2\pi$, then we would have $w(\beta, 0) = -1$. If we extended the domain of γ so that $0 \le t \le 4\pi$, then we would have $w(\gamma, 0) = 2$, since γ now winds twice around the origin.

Proposition 2.2.11. *Let γ be a closed contour that does not pass through $z_0 \in \mathbb{C}$. Then*

$$w(\gamma, z_0) = \frac{1}{2\pi i}\int_\gamma \frac{1}{z - z_0}\,dz.$$

Proof. Omitted. □

Theorem 2.2.12 (Generalised Cauchy Theorem)**.** *Let D be a domain and f holomorphic on D. Let $\gamma_1, \dots, \gamma_n$ be closed contours in D. Let $w(\gamma_i, z)$ be the winding number of γ_i around z. Suppose that $w(\gamma_1, z) + \cdots + w(\gamma_n, z) = 0$ for all $z \notin D$. Then*

$$\int_{\gamma_1} f + \cdots + \int_{\gamma_n} f = 0.$$

Proof. Omitted. □

Theorem 2.2.13 (Cauchy's Integral Formula)**.** *Suppose f is holomorphic on the disc $D = \{z \in \mathbb{C} \mid |z - z_0| < R\}$. For $0 < r < R$, let C_r be the path $C_r(t) = z_0 + re^{it}$, $0 \le t \le 2\pi$ (the circle with centre z_0 and radius r). Then for all $w \in \mathbb{C}$ such that $|w - z_0| < r$, we have:*

$$f(w) = \frac{1}{2\pi i} \int_{C_r} \frac{f(z)}{z - w}\, dz.$$

Proof. We omit some details for brevity. Fix w and consider

$$g(z) = \frac{f(z) - f(w)}{z - w}.$$

Then g is differentiable on $D \setminus \{w\}$. Define S_ε to be the circle centred at w of radius $\varepsilon > 0$. We take ε to be sufficiently small such that both C_r and S_ε are contained in D, and then apply the Generalised Cauchy Theorem (2.2.12) to the function g and paths $-C_r$ and S_ε, giving

$$\int_{S_\varepsilon} g = \int_{C_r} g. \tag{2.1}$$

Noting that $\lim_{z \to w} g(z) = f'(w)$ and that $|f'(w)|$ is finite, we have that $|g(z)|$ is bounded from above by some M for all z suffiently close to w. So by the Estimation Lemma (2.2.8) and equation (2.1), we have

$$\left| \int_{C_r} g \right| \le 2\pi M \varepsilon.$$

We can take ε to be arbitrarily close to 0, so in fact we have

$$\int_{C_r} g = 0.$$

The result follows from Proposition 2.2.11 and the fact that $f(w)$ is constant. □

This is a surprising result; it tells us that if we know the value of a function along the boundary of a circle, then we know its value at any point inside that circle. It is worth noting that this is unique to complex analysis, i.e. there is no analogous result for circles in $\mathbb{R}^n$.

2.3 Banach Spaces & Bounded Linear Operators

To aid intuition, we stick to familiar examples throughout this section, focusing on others only when they are relevant to later work. A more complete treatment, including omitted proofs, can be found in [Bol90].

Definition 2.3.1. A *normed vector space* is a vector space V over a field $\mathbb{K} \subseteq \mathbb{C}$ on which a *norm* is defined. That is, we have a map $|| \cdot || : V \to \mathbb{R}$ such that for all $x,\, y \in V$, $\lambda \in \mathbb{K}$:

(i) $||x|| \geq 0$;

(ii) $||x|| = 0 \iff x = \underline{0}$;

(iii) $||\lambda x|| = \lambda ||x||$;

(iv) $||x + y|| \leq ||x|| + ||y||$ (the Triangle Inequality).

Remark 2.3.2. A norm induces a metric on a vector space by

$$\mathrm{dist}(x,\, y) = ||x - y||.$$

Hence, a normed vector space is a metric space with the above metric.

Remark 2.3.3. Any two norms $|| \cdot ||_1$ and $|| \cdot ||_2$ on a finite-dimensional vector space V are *equivalent* if there exist $C_1,\, C_2 \in \mathbb{R}$ with $0 < C_1 \leq C_2$ such that for all $x \in V$

$$C_1 ||x||_2 \leq ||x||_1 \leq C_2 ||x||_2.$$

Example 2.3.4. For $\mathbf{x} \in \mathbb{R}^n$, we can define the *Euclidean norm* $||\mathbf{x}|| = \sqrt{x_1^2 + \cdots + x_n^2}$. Thus, $\mathbb{R}^n$ is a normed vector space with the Euclidean norm. This is our standard notion of vector magnitude in Euclidean space. It is also clear that this norm induces our standard distance metric $d(\mathbf{x},\, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||$ on $\mathbb{R}^n$. By Remark 2.3.3, any other norm we may define on $\mathbb{R}^n$ is equivalent to the Euclidean norm.

Definition 2.3.5. A sequence $x_1,\, x_2, \ldots$ of points in a metric space $(X,\, d)$ is called *Cauchy* if for every real number $\varepsilon > 0$, there exists a natural number N such that for every pair of integers $n,\, m > N$, $\mathrm{dist}(x_n, x_m) < \varepsilon$. In other words, for each $\varepsilon > 0$, all but finitely many points in the sequence are a distance of less than ε apart.

Definition 2.3.6. A metric space X is called *complete* if every Cauchy sequence of points in X has a limit in X.

Example 2.3.7. $\mathbb{R}^n$ is a complete metric space (see [Bry85] p. 41 for a proof of this for $\mathbb{R}^2$, which extends naturally to $\mathbb{R}^n$).

Definition 2.3.8. A *Banach space* is a complete normed vector space.

Example 2.3.9. $\mathbb{R}^n$ is a Banach space, by Examples 2.3.4 and 2.3.7.

Definition 2.3.10. Let $V,\, W$ be normed vector spaces. We call a linear operator $L : V \to W$ *bounded* if there exists some $M > 0$ such that for all $v \in V$

$$||Lv|| \leq M||v||. \tag{2.2}$$

We define the *operator norm* of a bounded linear operator L, $||L||$, as the smallest M such that equation (2.2) holds. Equivalently, we can write

$$||L|| = \sup_{||v||=1} ||Lv||.$$

That is, the operator norm is the maximum value by which L stretches an element of V.

Example 2.3.11. Recall that the linear maps on $\mathbb{R}^n$ are precisely the $n \times n$ matrices with real entries. Since all the entries in such a matrix are finite, all such matrices are clearly bounded. Hence any $n \times n$ matrix with real entries is a bounded linear operator on $\mathbb{R}^n$. Note that, since we can think of $M_n(\mathbb{R})$ as a vector space over $\mathbb{R}$ with dimension n^2, the operator norm on $M_n(\mathbb{R})$ is equivalent to any other norm on this space, by Remark 2.3.3.

Example 2.3.12. Given normed spaces X, Y over the same field $\mathbb{K} \subseteq \mathbb{C}$, denote by $B(X,Y)$ the set of all continuous $\mathbb{K}$-linear maps $L : X \to Y$. It is straightforward to check that this is a vector space. Hence, it is a normed vector space under the operator norm. If Y is a Banach space, then $B(X,Y)$ is also Banach (see [Bol90], p. 30-1).

Example 2.3.13. We are particularly interested in spaces of functions such as in the previous example. Fix a domain $D \subseteq \mathbb{C}$ and consider the space $\mathcal{A} = A_\infty(D) = \{f \mid f \text{ is holomorphic on } D \text{ and has a continuous extension to the closure of } D\}$. Then $\mathcal{A}$ is Banach. In section 3.2 we restrict a linear operator to the space $\mathcal{A}$ (for a specific domain D); this allows us to express the operator as a tensor product, and in turn prove that it is nuclear (see section 2.4).

Theorem 2.3.14. *A linear operator between normed vector spaces is bounded if and only if it is continuous.*

Proof. From [Bol90], p. 28-9.

($\Longrightarrow$) Suppose $L : X \to Y$ is bounded, then $||Lx|| \leq N||x||$ for all $x \in X$. Then, if $||x - y|| < \frac{\varepsilon}{N}$, we have $||Lx - Ly|| < \varepsilon$, and L is continous.

($\Longleftarrow$) Suppose $L : X \to Y$ is continuous, and fix $x_0 \in X$. Then $B(Lx_0, 1)$ is a neighbourhood of Lx_0, so there exists a $\delta > 0$ such that

$$x = x_0 + y \in B(x_0, \delta) \implies Lx = Lx_0 + Ly \in B(Lx_0, 1).$$

In other words, if $||y|| \leq \delta$, then $||Ly|| \leq 1$. So $||Lz|| \leq \frac{1}{\delta}||z||$ for all $z \in X$, and L is bounded. □

Definition 2.3.15. Given a bounded linear operator $L : X \to Y$, the *spectrum* of L is defined by

$$\sigma(L) = \{\lambda \in \mathbb{C} \mid (L - \lambda I) \text{ is not bijective}\}.$$

Remark 2.3.16. We think of the spectrum of an operator as a generalisation of the set of eigenvalues of a matrix. In general, $\sigma(L)$ contains all the eigenvalues of L, but may contain other values.

Definition 2.3.17. Let L be a bounded linear operator on a Banach space with spectrum $\sigma(L)$. Then the *spectral radius* of L is given by

$$\rho(L) = \sup_{\lambda \in \sigma(L)} |\lambda|.$$

Example 2.3.18. As we may expect, the spectrum of a square matrix M with real entries is precisely the set of eigenvalues of M. If M has eigenvalues $\lambda_1, \dots, \lambda_n$, then the spectral radius of M is given by

$$\rho(M) = \max_i\{|\lambda_i|\},$$

i.e. it is the greatest magnitude of any eigenvalue of M.

Theorem 2.3.19 (Spectral Radius Formula)**.** *Let L be a bounded linear operator on a Banach space, and let $\rho(L)$ be the spectral radius of L. Then*

$$\rho(L) = \lim_{k\to\infty} ||L^k||^{\frac{1}{k}},$$

where $||\cdot||$ is the operator norm.

Proof. Omitted. □

2.4 Nuclear & Trace-Class Operators

We aim to generalise the notion of a matrix trace to bounded linear operators on general Banach spaces, following the topic introduced by Grothendieck in [Gro54], as presented by Mayer in the appendix of [May91]. We continue with the same examples from Section 2.3.

In this section, arbitrary vector and Banach spaces are taken over $\mathbb{C}$.

Definition 2.4.1. Let V be a vector space over $\mathbb{C}$. A *linear functional* on V is a linear map $\alpha : V \to \mathbb{C}$. The *dual space* V^* of V is the set of all linear functionals on V, endowed with the vector space structure of addition and scalar multiplication.

Remark 2.4.2. Depending on the context, linear functionals may be called *linear forms*, *one-forms*, or *covectors.*

Remark 2.4.3. V^* is a normed vector space under the operator norm.

Example 2.4.4. Let $V = \mathbb{R}^n$. Then $(\mathbb{R}^n)^* = \{\alpha : \mathbb{R}^n \to \mathbb{R} \mid \alpha \text{ is linear}\}$. For example, take $\alpha(\mathbf{x}) = e_1 \cdot \mathbf{x}$, where e_1 is the first standard basis vector of $\mathbb{R}^n$. Then α corresponds to projection onto e_1, which in particular is linear on $\mathbb{R}^n$, so $\alpha \in (\mathbb{R}^n)^*$.

In particular, we may consider linear functionals on a Banach space B, which form the dual of the space, B^*. Recall that for $e \in B$, $\alpha \in B^*$, the tensor $\alpha \otimes e$ is a linear map $B \to B$ defined by $(\alpha \otimes e)(v) = \alpha(v)e$. We now consider the tensor product space $B^* \bigotimes B$, which consists of all elements of the form $\mathcal{L} = \sum_i \lambda_i \alpha_i \otimes e_i$ with $e_i \in B$, $\alpha_i \in B^*$, $\lambda_i \in \mathbb{C}$ and $||e_i|| = ||\alpha_i|| = 1$. Such an $\mathcal{L}$ acts on elements $v \in B$ by

$$\mathcal{L}v = \sum_i \lambda_i(\alpha_i \otimes e_i)(v) = \sum_i \lambda_i \alpha_i(v) e_i.$$

Since the α_i are linear, $\mathcal{L}$ is a linear map $B \to B$.

Note that representations of $\mathcal{L}$ are not unique; that is, there is no unique way to choose the e_i, α_i, λ_i such that $\mathcal{L} = \sum_i \lambda_i \alpha_i \otimes e_i$.

Example 2.4.5. Take $B = \mathbb{R}^n$. Then $(\mathbb{R}^n)^* \bigotimes \mathbb{R}^n$ is the space of all linear maps $\mathbb{R}^n \to \mathbb{R}^n$. That is, it is the space of all $n \times n$ matrices with real entries. We investigate a concrete example in the two-dimensional case; let e_1, e_2 be the standard basis vectors on $\mathbb{R}^2$, let $\mathbf{x} = x_1e_1 + x_2e_2 \in \mathbb{R}^2$, and consider

$$A = \begin{pmatrix} 2 & 0 \\ 0 & 5 \end{pmatrix}$$

Then $A\mathbf{x} = 2x_1e_1 + 5x_2e_2$. The matrix A is diagonal, so clearly the eigenvalues of A are 2 and 5, with corresponding eigenvectors e_1 and e_2. Let α_1, α_2 be the projection maps onto e_1, e_2, respectively (explicitly, $\alpha_i(\mathbf{x}) = e_i \cdot \mathbf{x} = x_i$). Then we may write

$$\begin{aligned} A\mathbf{x} &= 2x_1e_1 + 5x_2e_2 \\ &= 2\alpha_1(\mathbf{x})e_1 + 5\alpha_2(\mathbf{x})e_2 \\ &= \sum_{i=1}^{2} \lambda_i \alpha_i(\mathbf{x})e_i \\ &= \sum_{i=1}^{2} \lambda_i(\alpha_i \otimes e_i)(\mathbf{x}). \end{aligned}$$

Note that $||\alpha_i|| = ||e_i|| = 1$ for $i = 1, 2$. Hence A is indeed an element of the space $(\mathbb{R}^2)^* \bigotimes \mathbb{R}^2$.

Definition 2.4.6. A bounded linear operator $\mathcal{L} : B \to B$ is *nuclear* and of *class-p* $(0 < p \leq 1)$ if it has a representation

$$\mathcal{L} = \sum_i \lambda_i \alpha_i \otimes e_i$$

where $e_i \in B$, $\alpha_i \in B^*$, $\lambda_i \in \mathbb{C}$, such that $||e_i|| = ||\alpha_i|| = 1$ and $\sum_i |\lambda_i|^p < \infty$.

Definition 2.4.7. The *order* of a nuclear operator $\mathcal{L}$ is the infimum of all $0 < p \leq 1$ such that $\mathcal{L}$ is class-p.

Example 2.4.8. All $n \times n$ matrices with real (or complex) entries are nuclear of order 0; they have only finitely-many eigenvalues, so $\sum_{i=1}^n |\lambda_i|^p < \infty$ for all $0 < p \leq 1$.

Remark 2.4.9. Any bounded linear operator on a finite-dimensional Banach space is nuclear of order zero, as the sum $\sum_i |\lambda_i|^p < \infty$ for all $0 < p \leq 1$. Hence, the distinction is only important in the case of infinite-dimensional spaces, such as the function spaces with which we are concerned.

Theorem 2.4.10 (Grothendieck Trace Theorem). *Let $\mathcal{L} : B \to B$ be nuclear of order $\leq \frac{2}{3}$ with eigenvalues λ_i (counted with multiplicities). Then $\mathcal{L}$ has a trace, given by*

$$\mathrm{tr}(\mathcal{L}) = \sum_i \lambda_i.$$

Define the ***Fredholm determinant*** $\det(I - z\mathcal{L})$*, a function of z which is holomorphic on $\mathbb{C}$, by*

$$\det(I - z\mathcal{L}) = \prod_i (1 - z\lambda_i).$$

Then the following formula for the Fredholm determinant holds

$$\det(I - z\mathcal{L}) = \exp(\operatorname{tr}(\log(I - z\mathcal{L}))).$$

From the definition of the Fredholm determinant, we have

$$\det(I - z\mathcal{L}) = 0 \iff z = \frac{1}{\lambda_i} \text{ for some } i \geq 1.$$

In other words, the zeroes of the Fredholm determinant of a nuclear operator $\mathcal{L}$ (counted with multiplicities) are precisely the reciprocals of the non-zero eigenvalues of $\mathcal{L}$ (counted with multiplicities). This is a standard result for $n \times n$ matrices, which we have now extended to nuclear operators on general Banach spaces. The application of this result to the transfer operator in section 3.2 is a crucial step in the proof of one of the major results of this paper.

2.5 Transfer Operators

Definition 2.5.1. Given a map $T : [0,\,1] \to [0,\,1]$, we define the *transfer operator* of T by

$$\mathcal{L}_g w(x) = \sum_{y \,|\, Ty = x} e^{g(y)}\, w(y),$$

where $g(x)$ is some *weight function.* Note that the sum is taken over all y such that $Ty = x$; that is, it is taken over the preimage of x under T. The transfer operator is sometimes called the *Ruelle-Perron-Frobenius operator.*

Remark 2.5.2. For our purposes, we take $g(x) = -s \log |T'(x)|$, giving

$$\mathcal{L}_s w(x) = \sum_{y \,|\, Ty = x} \frac{1}{|T'(y)|^s}\, w(y).$$

Note that $\mathcal{L}_g$ acts on the (Banach) space of functions $[0,\,1] \to \mathbb{C}$. It is clearly continuous, hence it is bounded by Theorem 2.3.14, and so we can apply our theory from Section 2.3. In particular, we are interested in the spectral properties of $\mathcal{L}_s$.

In the following examples, we fix $s = 1$.

Example 2.5.3. For a straightforward example, let us consider the transfer operator of the *doubling map* $T : [0,1] \to [0,1]$, $Tx = 2x \mod 1$. The preimage of x under T is the set $\{\frac{x}{2},,\ \frac{x+1}{2}\}$, and we have $T'(y) = 2$ for all points y. Therefore, our transfer operator is given by

$$\mathcal{L}_s w(x) = \frac{1}{2} w\left(\frac{x}{2}\right) + \frac{1}{2} w\left(\frac{x+1}{2}\right).$$

We may consider iterating a general transfer operator:

$$\begin{aligned}
\mathcal{L}_g^2 w(x) = \mathcal{L}_g(\mathcal{L}_g)(x) &= \sum_{y|Ty=x} e^{g(y)} \mathcal{L}_g w(y) \\
&= \sum_{y|Ty=x} e^{g(y)} \sum_{z|Tz=y} e^{g(z)} w(z) \\
&= \sum_{z|T^2z=x} e^{g(Tz)} e^{g(z)} w(z).
\end{aligned}$$

In general, for $n \in \mathbb{N}$, we have

$$\mathcal{L}_g^n w(x) = \sum_{y | T^n y = x} e^{g(y)+g(Ty)+\cdots+g(T^{n-1}y)} w(y).$$

When $g(x) = -s \log |T'(x)|$, we obtain

$$\begin{aligned} \mathcal{L}_s^n w(x) &= \sum_{y|T^n y = x} (|T'(y)| \cdot |T'(Ty)| \cdots |T'(T^{n-1}y)|)^{-s} w(y) \\ &= \sum_{y|T^n y=x} \frac{1}{|(T^n)'(x)|^s} w(y) \end{aligned} \tag{2.3}$$

by the chain rule.

Example 2.5.4. Iterating our operator $\mathcal{L}_s$ from Example 2.5.3, we get

$$\mathcal{L}_s^2 w(x) = \frac{1}{4} w\left(\frac{x}{4}\right) + \frac{1}{4} w\left(\frac{x+1}{4}\right) + \frac{1}{4} w\left(\frac{x+2}{4}\right) + \frac{1}{4} w\left(\frac{x+3}{4}\right),$$

and in general, we have

$$\mathcal{L}_s^n w(x) = \sum_{j=0}^{2^n-1} \frac{1}{2^n} w\left(\frac{x+j}{2^n}\right).$$

This motivates us to look at the limiting behaviour, and we see that

$$\lim_{n\to\infty} \mathcal{L}_s^n w(x) = \int w(x) dx.$$

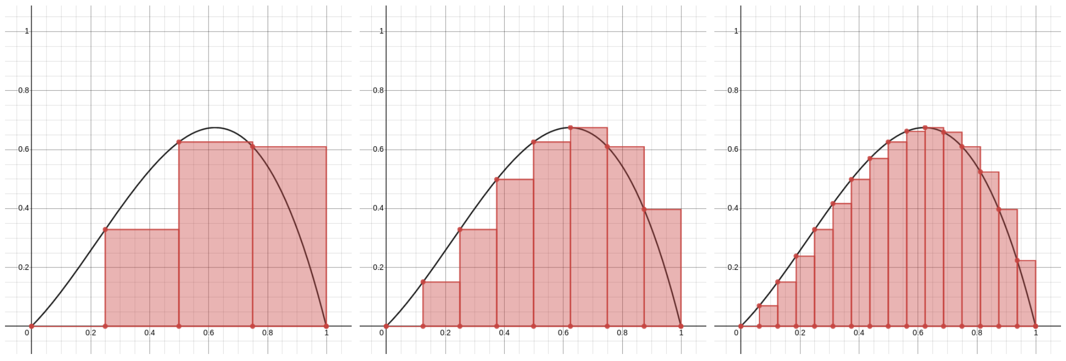

Figure 2: Visual intuition for the limiting behaviour of $\mathcal{L}_s^n$, showing that it approaches an integral as $n \to \infty$.

Example 2.5.5. As a precursor to the work that follows, we investigate the transfer operator of the *Gauss map* $T : [0,1] \to [0,1]$, $Tx = \frac{1}{x} \mod 1$. The preimage of x under T is the set of numbers of the form $y = \frac{1}{n+x}$, with $n \in \mathbb{N}$. Note that we have $|T'(y)| = \frac{1}{y^2} = (n+x)^2$, so our transfer operator is given by

$$\mathcal{L}_s w(x) = \sum_{n=1}^{\infty} \left(\frac{1}{n+x}\right)^2 w\left(\frac{1}{n+x}\right).$$

Now, let $f(x) = \frac{1}{\log 2}\frac{1}{1+x}$. We claim that f is an eigenfunction of $\mathcal{L}_s$, and aim to find its corresponding eigenvalue λ_1. We have

$$\begin{aligned}
\mathcal{L}_s f(x) &= \sum_{n=1}^{\infty} \left(\frac{1}{n+x}\right)^2 \frac{1}{\log 2}\frac{1}{1+\frac{1}{n+x}} = \frac{1}{\log 2}\sum_{n=1}^{\infty} \frac{1}{(n+x)^2}\frac{1}{1+\frac{1}{n+x}} \\
&= \frac{1}{\log 2}\sum_{n=1}^{\infty} \frac{1}{(n+x)^2+n+x} = \frac{1}{\log 2}\sum_{n=1}^{\infty} \frac{1}{n^2+n+(2n+1)x+x^2} \\
&= \frac{1}{\log 2}\sum_{n=1}^{\infty} \frac{1}{(n+x)(n+1+x)}.
\end{aligned}$$

Using partial fraction decomposition, we can show that

$$\frac{1}{(n+x)(n+1+x)} = \frac{1}{n+x} - \frac{1}{n+1+x}.$$

Hence,

$$\begin{aligned}
\mathcal{L}_s f(x) &= \frac{1}{\log 2}\left(\sum_{n=1}^{\infty}\frac{1}{n+x} - \sum_{n=1}^{\infty}\frac{1}{n+1+x}\right) \\
&= \frac{1}{\log 2}\left(\sum_{n=1}^{\infty}\frac{1}{n+x} - \sum_{n=2}^{\infty}\frac{1}{n+x}\right) \\
&= \frac{1}{\log 2}\frac{1}{1+x}.
\end{aligned}$$

So f is an eigenfunction of $\mathcal{L}_s$, as claimed, and its corresponding eigenvalue is $\lambda_1 = 1$. In fact, this is the largest eigenvalue of $\mathcal{L}_s$ [May91].

3 Main Results

3.1 Statement of Main Results

Recall that, by the Dimension Formula (Theorem 2.1.6), we aim to find the unique s such that $P(-s\log|T'|)=0$. From the definition of pressure, we have

$$\begin{aligned} P(-s\log|T'|) &= \lim_{n\to\infty}\frac{1}{n}\log\left(\sum_{T^n x=x}\exp\left(-s\log|T'(x)|-s\log|T'(Tx)|-\cdots-s\log|T'(T^{n-1}x)|\right)\right) \\ &= \lim_{n\to\infty}\frac{1}{n}\log\left(\sum_{T^n x=x}\left(|T'(x)|\cdot|T'(Tx)|\cdots|T'(T^{n-1}x)|\right)^{-s}\right). \end{aligned} \tag{3.1}$$

Hence, we can express pressure using terms of the form $|T'(T^k x)|^{-s}$. Note the similarity to equation 2.3. Now, here comes the trick: notice that $|T'(x)|=\frac{1}{x^2}=(T(x))^2$. This means we can write

$$|T'(T^k x)|^{-s}=(T^k x)^{2s}.$$

This is an important but subtle step; in general, T' cannot be written in terms of T. The fact that we can do this for the Gauss map is the crucial difference that means the problem is somewhat tractable in this case.

In the pressure formula, we are summing over the points x such that $T^n x=n$, i.e. the periodic points of order n. Due to the nature of the Gauss map, we are able to find a tidy formula for these points. First, we take a sequence $\underline{i}=i_1,\ldots,i_n\in\{2,\,5\}^n$. Then we define

$$\begin{aligned} T_{\underline{i}} &= T_{i_1}\circ\cdots\circ T_{i_n}:[0,\,1]\to[0,\,1], \\ T_{\underline{i}} &: x\mapsto \cfrac{1}{i_1+\cfrac{1}{i_2+\cdots\cfrac{1}{i_n+x}}}. \end{aligned}$$

It is clear that the unique fixed point of $T_{\underline{i}}$ is the periodic continued fraction $[\overline{i_1,\ldots,i_n}]$, which we denote $x_{\underline{i}}$. We say that $T_{\underline{i}}$ is *locally inverse* to T^n, in that $T_{\underline{i}}(T^n(x_{\underline{i}}))=x_{\underline{i}}$. We consider the orbit of the $x_{\underline{i}}$ under T, which is the collection $\{x_{\underline{i}},\,Tx_{\underline{i}},\ldots,T^{n-1}x_{\underline{i}}\}$.

Remark 3.1.1. This is a slight abuse of notation, as we actually consider this orbit **with multiplicities**. To see why this is a concern, consider the sequence 2525 with corresponding periodic point x_{2525} of period 4. We can see, however, that x_{2525} also has period 2, so its 4-orbit will contain duplicates of each point. In general, a point of period k also has period $2k$, $3k$, etc.. This is relevant since we take the sum over all sequences of length n (as opposed to those of length n with period n). We will see that the practical implications of this are minimal, due to some nice properties of sequence concatenation (see section 4.2).

Denote by $|\underline{i}|$ the length of the sequence $\underline{i}$. Now, following the notation used in [JP01], we define the *weight* of $\underline{i}$ by

$$w_{\underline{i}}=\prod_{k=0}^{|\underline{i}|-1}T^k x_{\underline{i}}.$$

Remark 3.1.2. Notice that $T^k x_{\underline{i}}=[\overline{i_{k+1},\ldots,i_n,\,i_1,\ldots,i_k}]$ for $k=0,\ldots,n-1$. This identity will prove helpful in calculating the weights.

We can now write

$$P(-s\log|T'|) = \lim_{n\to\infty}\frac{1}{n}\log\left(\sum_{|\underline{i}|=n} w_{\underline{i}}^{2s}\right)$$

by equation (3.1) and the fact that the $x_{\underline{i}}$ with $|\underline{i}| = n$ are precisely the periodic points of period n (all x with $T^n x = x$).

Recall from Remark 2.5.2 that we take $\mathcal{L}_s w(x) = \sum_{y\,|\,Ty=x}\frac{1}{|T'(y)|^s}\,w(y)$. We have $|T'(y)| = (T(y))^2 = \frac{1}{y^2}$, hence for our restricted Gauss map T we have

$$\mathcal{L}_s w(x) = \left(\frac{1}{2+x}\right)^{2s} w\left(\frac{1}{2+x}\right) + \left(\frac{1}{5+x}\right)^{2s} w\left(\frac{1}{5+x}\right).$$

We are now ready to state the main results of this paper. We present the statement of the main results as given in [JP01], modified to reflect that we are interested in $E_{2,5}$ instead of $E_{1,2}$. Note also that our definition of $b_n(s)$ differs from that given in Jenkinson & Pollicott's paper (the factor of $\frac{2}{n_j}$ is omitted). This is due to a presumed typo in that paper.

Theorem 3.1.3. *Let*

$$b_n(s) = \begin{cases} 0 & \text{if } n \text{ is odd,} \\ \sum_{r=1}^{\infty}\sum_{n_1+\dots+n_r=\frac{n}{2}}\frac{(-1)^r}{r!}\prod_{j=1}^{r}\frac{2}{n_j}\left(\sum_{|\underline{i}|=2n_j}\frac{w_{\underline{i}}^{2s}}{1-w_{\underline{i}}^2}\right) & \text{if } n \text{ is even.}\end{cases}$$

Then

(a) The Hausdorff dimension of $E_{2,5}$ is the largest solution $0 \le s \le 1$ to the equation

$$D(s,1) = \det(I - \mathcal{L}_s^2) = 1 + \sum_{n=1}^{\infty} b_n(s) = 0.$$

(b) There exist $0 < \theta < 1$ and $K > 0$ such that if s_N denotes the largest solution $0 \le s_N \le 1$ to the equation

$$1 + \sum_{n=1}^{N} b_n(s) = 0,$$

then $|s_N - \dim_H E_{2,5}| < K\theta^{N^2}$.

3.2 Proof of (a)

We must first convince ourselves that the Fredholm determinant $\det(I - z\mathcal{L}_s)$ is defined. Hence, we aim to show that $\mathcal{L}_s$ is nuclear. The first step is to restrict the transfer operator $\mathcal{L}_s$ to a suitable Banach space of functions on a disk in the complex plane.

We define the open disk

$$D = \{z \in \mathbb{C} \mid |z-1| < \tfrac{3}{2}\}.$$

Then the images of D under the maps T_2 and T_5 are given by

$$T_2(D) = \{z \in \mathbb{C} \mid |z - \tfrac{4}{9}| < \tfrac{2}{9}\},$$
$$T_5(D) = \{z \in \mathbb{C} \mid |z - \tfrac{8}{45}| < \tfrac{2}{45}\}.$$

Note that the closure of the image disks $T_2(D)$, $T_5(D)$ lie strictly within D.

We restrict $\mathcal{L}_s$ to the space $\mathcal{A}$ of analytic functions on D which have a continuous extension to the closure of D. This is indeed a Banach space under the operator norm by Example 2.3.13.

Next, define the curve

$$\Gamma = \{z \in \mathbb{C} \mid |z - \tfrac{10}{9}| = 1\}.$$

Note that Γ is strictly within D, and does not intersect either of the image disks $T_2(D)$, $T_5(D)$.

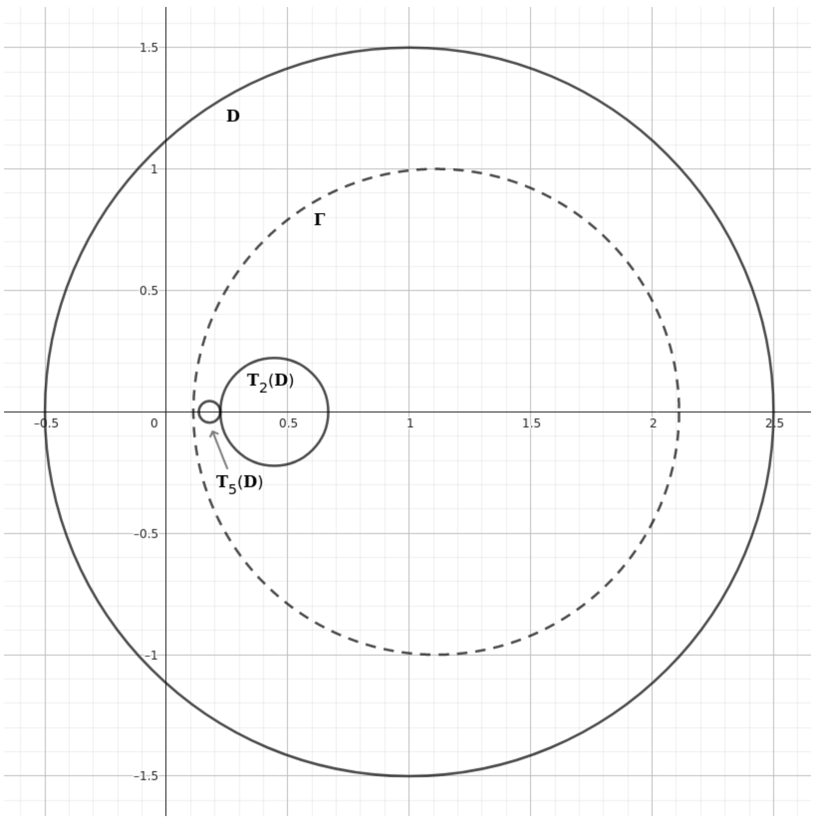

Figure 3: The disk D, its images under the maps T_2 and T_5, and the curve Γ.

Let $v \in \mathcal{A}$, and $f(z) = z^{2s}v(z)$, and note that $\mathcal{L}_s v(z) = f(T_2 z) + f(T_5 z)$. Notice also that $g(z) = z^{2s}$ is well-defined and holomorphic on the region enclosed by the curve Γ. Hence, f is also holomorphic on this region, and we can apply Cauchy's Integral Formula (Theorem 2.2.13), giving

$$f(T_j z) = \frac{1}{2\pi i} \int_\Gamma \frac{\xi^{2s} v(\xi)}{\xi - T_j z}\, d\xi.$$

Hence

$$\begin{aligned}
\mathcal{L}_s v(z) &= \frac{1}{2\pi i}\int_\Gamma \xi^{2s} v(\xi)\left(\frac{1}{\xi - T_2 z} + \frac{1}{\xi - T_5 z}\right) d\xi \\
&= \frac{1}{2\pi i}\int_\Gamma \xi^{2s} v(\xi)\left(\frac{1}{(\xi-\frac{10}{9}) - (T_2 z - \frac{10}{9})} + \frac{1}{(\xi-\frac{10}{9}) - (T_5 z - \frac{10}{9})}\right) d\xi \\
&= \frac{1}{2\pi i}\int_\Gamma \frac{\xi^{2s} v(\xi)}{\xi - \frac{10}{9}}\left(\frac{1}{1-\frac{T_2 z - \frac{10}{9}}{\xi - \frac{10}{9}}} + \frac{1}{1-\frac{T_5 z - \frac{10}{9}}{\xi - \frac{10}{9}}}\right) d\xi \\
&= \frac{1}{2\pi i}\int_\Gamma \frac{\xi^{2s} v(\xi)}{\xi - \frac{10}{9}}\sum_{n=0}^{\infty}\left(\left(\frac{T_2 z - \frac{10}{9}}{\xi - \frac{10}{9}}\right)^n + \left(\frac{T_5 z - \frac{10}{9}}{\xi - \frac{10}{9}}\right)^n\right) d\xi \\
&= \sum_{n=0}^{\infty}\left(\left(T_2 z - \frac{10}{9}\right)^n + \left(T_5 z - \frac{10}{9}\right)^n\right)\left(\frac{1}{2\pi i}\int_\Gamma \frac{\xi^{2s} v(\xi)}{(\xi - \frac{10}{9})^{n+1}}\, d\xi\right)
\end{aligned}$$

by the geometric series formula

$$\sum_{n=0}^{\infty} r^n = \frac{1}{1-r}$$

for $|r| < 1$. Note that we can apply this formula here, since Γ is outside the image disks $T_2(D)$, $T_5(D)$, so

$$\left|\frac{T_j z - \frac{10}{9}}{\xi - \frac{10}{9}}\right| < 1$$

for all $z \in D$, $\xi \in \Gamma$.

We are now encouragingly close to the required form for $\mathcal{L}_s$ to be a bounded linear operator on $\mathcal{A}$, as outlined in Section 2.4. The next steps are straightforward bookkeeping; we may now write

$$\begin{aligned}
g_{j,m}(z) &= \left(T_j z - \frac{10}{9}\right)^m, \\
l_m(v) &= \frac{1}{2\pi i}\int_\Gamma \frac{\xi^{2s} v(\xi)}{(\xi - \frac{10}{9})^{m+1}}\, d\xi.
\end{aligned}$$

To simplify indices, set

$$k = k(j) = \begin{cases} 1 & \text{if } j = 2 \\ 2 & \text{if } j = 5 \end{cases}$$

and define the normalisations

$$\begin{aligned}
\hat{g}_{2m+k} &= \frac{g_{j,m}}{||g_{j,m}||} \\
\hat{l}_{2m+k} &= \frac{l_m}{||l_m||}
\end{aligned}$$

for $m \geq 0$, $k = 1, 2$. Then we can write

$$\mathcal{L}_s v(z) = \sum_{n=1}^{\infty} \lambda_n \hat{l}_n(v) \hat{g}_n(z)$$

where $\lambda_{2m+k} = ||g_{j,m}|| \cdot ||l_m||$. Hence, we have written $\mathcal{L}_s$ as an element of $\mathcal{A}^* \bigotimes \mathcal{A}$. That is, $\mathcal{L}_s$ is a bounded linear operator on $\mathcal{A}$. To show that it is nuclear, it suffices to show that $\sum_{n=1}^{\infty} |\lambda_n| < \infty$.

We have $\lambda_{2m+k} = ||g_{j,m}|| \cdot ||l_m|| \geq 0$, so we estimate the norms $||g_{j,m}||$ and $||l_m||$. Let $\xi \in \Gamma$, then $|\xi - \frac{10}{9}| = 1$ and in turn $|\xi^{2s}| \leq |\frac{19}{9}|^{2s} < 5$ (for $0 \leq s \leq 1$). Recall that $||l_m|| = \sup_{||v||=1} |l_m(v)|$, so we may take $||v(\xi)|| \leq 1$. Then we have

$$\left| \frac{\xi^{2s} v(\xi)}{(\xi - \frac{10}{9})^{m+1}} \right| = \frac{|\xi^2 s| \cdot |v(\xi)|}{|\xi - \frac{10}{9}|^m m + 1} \leq \frac{5 \cdot 1}{1} = 5.$$

So, by the Estimation Lemma (2.2.8),

$$||l_m|| \leq \left| \frac{1}{2\pi i} \int_{\Gamma} \frac{\xi^{2s} v(\xi)}{(\xi - \frac{10}{9})^{m+1}} \right| \leq \frac{5}{2\pi} \cdot \text{length}(\Gamma) = \frac{5}{2\pi} \cdot 2\pi = 5.$$

Now, let $z \in D$. We have $|T_2 z - \frac{10}{9}|$, $|T_5 z - \frac{10}{9}| \leq \frac{44}{45}$, so $||g_{j,m}|| \leq \left(\frac{44}{45}\right)^m$.

So $\lambda_{2m+k} \leq 5 \cdot \left(\frac{44}{45}\right)^m = \frac{225}{44} \left(\sqrt{\frac{44}{45}}\right)^{2m+2}$ for $m \geq 0$, $k = 1, 2$. Hence $\lambda_n \leq \frac{225}{44} \left(\sqrt{\frac{44}{45}}\right)^n$ for all $n \geq 1$. Therefore, in particular, the series $\sum_{n=1}^{\infty} |\lambda_n|$ converges.

Hence, $\mathcal{L}_s$ is nuclear.

By Theorem 2.4.10, we have that the zeroes of $\det(I - z\mathcal{L}_s)$ (counted with multiplicities) are precisely the reciprocals of the non-zero eigenvalues of $\mathcal{L}_s$ (counted with multiplicities).

Analogous to the formula for the logarithm of a square matrix, we have the following formula for the logarithm of a trace-class operator $\mathcal{L}$, from the Taylor series expansion for $\log x$:

$$\log \mathcal{L} = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{(\mathcal{L} - I)^k}{k},$$

if $||\mathcal{L} - 1|| < 1$. Setting $\mathcal{L} = I - z\mathcal{L}_s$, we have

$$\log(I - z\mathcal{L}_s) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{(-z\mathcal{L}_s)^k}{k} = -\sum_{k=1}^{\infty} \frac{z^k}{k} \mathcal{L}_s^k,$$

if $||\mathcal{L}_s|| < \frac{1}{|z|}$. So, by Theorem 2.4.10, we have

$$\begin{aligned} \det(I - z\mathcal{L}_s) &= \exp(\text{tr}(\log(I - z\mathcal{L}_s))) \\ &= \exp\left(\text{tr}\left(-\sum_{k=1}^{\infty} \frac{z^k}{k} \mathcal{L}_s^k\right)\right) \\ &= \exp\left(-\sum_{k=1}^{\infty} \frac{z^k}{k} \text{tr}\left(\mathcal{L}_s^k\right)\right) \end{aligned}$$

by additivity of the trace. Hence, we can find the non-zero eigenvalues of $\mathcal{L}_s$ by finding the reciprocals of zeroes of this expression. Using the power series for the exponential, and setting $a_k = \frac{\text{tr}(\mathcal{L}_s^k)}{k}$, we can express this as follows

$$\begin{aligned}\det(I - z\mathcal{L}_s) &= 1 + \sum_{n=1}^{\infty} \frac{(-1)^n}{n!} \left(\sum_{k=1}^{\infty} z^k a_k \right)^n \\ &= 1 + \sum_{n=1}^{\infty} \frac{(-1)^n}{n!} \sum_{k=1}^{\infty} z^k \sum_{k_1+\cdots+k_n=k} a_{k_1} \cdots a_{k_n} \\ &= 1 + \sum_{k=1}^{\infty} z^k \left(\sum_{n=1}^{\infty} \sum_{k_1+\cdots+k_n=k} \frac{(-1)^n}{n!} \prod_{j=1}^{n} a_{k_j} \right) \\ &= 1 + \sum_{n=1}^{\infty} z^n \left(\sum_{r=1}^{\infty} \sum_{n_1+\cdots+n_r=n} \frac{(-1)^r}{r!} \prod_{j=1}^{r} \frac{\operatorname{tr}(\mathcal{L}_s^{n_j})}{n_j} \right). \end{aligned} \tag{3.2}$$

We have the following result (see [PP90], p. 42-3).

Theorem 3.2.1. *Let β be the unique eigenvalue of maximum modulus of $\mathcal{L}_g$. Then $P(g) = \log \beta$.*

This tells us that the unique maximal eigenvalue of $\mathcal{L}_s$ is $e^{P(-s \log |T'|)}$. In turn, we have that the spectral radius of $\mathcal{L}_s$ is precisely $e^{P(-s \log |T'|)}$. Recall that by Theorem 2.1.6, $\dim_H E_{2,5}$ is the unique value of s for which $P(-s \log |T'|) = 0$. Hence, when $s = \dim_H E_{2,5}$, the unique maximal eigenvalue of $\mathcal{L}_s$ (and in turn its spectral radius) is precisely 1.

By the Spectral Radius Formula (Theorem 2.3.19), we have

$$\rho(-\mathcal{L}_s) = \lim_{k\to\infty} || - \mathcal{L}_s^k ||^{\frac{1}{k}} = \lim_{k\to\infty} ||\mathcal{L}_s^k||^{\frac{1}{k}} = \rho(\mathcal{L}_s).$$

Hence, the spectral radius of $-\mathcal{L}_s$ is also $e^{P(-s \log |T'|)}$.

Going forward, we consider $D(s, z) = \det(I - z\mathcal{L}_s)\det(I + z\mathcal{L}_s) = \det(I - z^s\mathcal{L}_s^2)$. We have $D(s, 1) = 0$ when $s = \dim_H E_{2,5}$. This provides improved computational performance, as it ensures that the odd terms in the power series expansion are zero (see section 4.1 for further details). However, this does not fundamentally change the nature of the work that follows, only some details.

It is clear that, when s is fixed, the zeroes of $D(s, z)$ are precisely the reciprocals of the non-zero eigenvalues of $\mathcal{L}_s$ and $-\mathcal{L}_s$ (all counted with multiplicities). We already have that when $s = \dim_H$, 1 is an eigenvalue of both $\mathcal{L}_s$ and $-\mathcal{L}_s$, so we can instead fix $z = 1$, and find $\dim_H E_{2,5}$ by calculating the values of s such that

$$D(s, 1) = \det(I - \mathcal{L}_s^2) = 0. \tag{3.3}$$

Note that we do not know that $\dim_H E_{2,5}$ is the only value of s for which 1 is an eigenvalue of $\mathcal{L}_s$ or $-\mathcal{L}_s$. However, suppose $s > \dim_H E_{2,5}$; then the spectral radius of both $\mathcal{L}_s$ and $-\mathcal{L}_s$ is less than 1, since $P(-s \log |T'|)$ is decreasing in s. So 1 cannot be an eigenvalue of either $\mathcal{L}_s$ or $-\mathcal{L}_s$, and it follows that $s = \dim_H E_{2,5}$ is in fact the largest solution to equation (3.3).

It remains to show that the terms $b_n(s)$, as given in the statement of the theorem, are the coefficients in the power series expansion of $D(s, 1)$. Looking at equation 3.2, we can see that we are now interested in finding the trace of $\mathcal{L}_s^n$ for arbitrary integers n.

Recall that $x_{\underline{i}}$ is the unique fixed point of the transformation $T_{\underline{i}} = T_{i_1} \circ \cdots \circ T_{i_n}$. That is, $T_{\underline{i}} x_{\underline{i}} = x_{\underline{i}}$. We also have that $x_{\underline{i}} = [\overline{i_1, \cdots, i_n}]$, and hence $T^k x_{\underline{i}} = [\overline{i_{k+1}, \cdots, i_n, i_1, \cdots i_k}]$.

We show that $T_{\underline{i}}'(x_{\underline{i}}) = (-1)^n w_{\underline{i}}^2$, where $n = |\underline{i}|$. Recall that $w_{\underline{i}} = \prod_{k=0}^{n-1} T^k x_{\underline{i}}$. By the chain rule, we have that

$$T_{\underline{i}}'(x) = \prod_{k=1}^{n} T_{i_k}'(T_{i_{k+1}} \circ \cdots \circ T_{i_n}(x)).$$

Note that $T_{i_{k+1}} \circ \cdots \circ T_{i_n}$ is locally inverse to T^{n-k}, and $(T^{n-k})^{-1} x_{\underline{i}} = T^k x_{\underline{i}} = [\overline{i_{k+1}, \cdots, i_n, i_1, \cdots i_k}]$. We also have that $T_{i_k}'(x) = \frac{-1}{(i_k+x)^2} = -(T_{i_k}(x))^2$. Hence we have

$$\begin{aligned}
T_{\underline{i}}'(x_{\underline{i}}) &= \prod_{k=1}^{n} T_{i_k}'(T^k x_{\underline{i}}) \\
&= \prod_{k=1}^{n} -(T_{i_k}([\overline{i_{k+1}, \cdots, i_n, i_1, \cdots i_k}]))^2 \\
&= \prod_{k=1}^{n} -([\overline{i_k, \cdots, i_n, i_1, \cdots i_{k-1}}])^2 \\
&= \prod_{k=1}^{n} -(T^{k-1} x_{\underline{i}})^2 = \prod_{k=0}^{n-1} -(T^k x_{\underline{i}})^2 \\
&= (-1)^n \left(\prod_{k=0}^{n-1} T^k x_{\underline{i}} \right)^2 = (-1)^n w_{\underline{i}}^2. \qquad (3.4)
\end{aligned}$$

Now, let $\mathcal{L}_{\underline{i}} v(z) = |T_{\underline{i}}'(z)|^s v(T_{\underline{i}} z)$, so that $\mathcal{L}_s^n = \sum_{|\underline{i}|=n} \mathcal{L}_{\underline{i}}$. Then by additivity of the trace we have that $\operatorname{tr}(\mathcal{L}_s^n) = \sum_{|\underline{i}|=n} \operatorname{tr}(\mathcal{L}_{\underline{i}})$. So it suffices to find the trace of $\mathcal{L}_{\underline{i}}$ for an arbitrary sequence $\underline{i}$.

Suppose λ is an eigenvalue of $\mathcal{L}_{\underline{i}}$ with corresponding eigenfunction v. Then $\mathcal{L}_{\underline{i}} v(z) = \lambda v(z)$ for all $z \in D$. In particular, we may take $z = x_{\underline{i}}$, giving

$$\begin{aligned}
|T_{\underline{i}}'(x_{\underline{i}})|^s v(T_{\underline{i}} x_{\underline{i}}) &= \lambda v(x_{\underline{i}}) \qquad (3.5) \\
w_{\underline{i}}^{2s} v(x_{\underline{i}}) &= \lambda v(x_{\underline{i}}),
\end{aligned}$$

by equation 3.4 and the fact that $x_{\underline{i}}$ is the unique fixed point of $T_{\underline{i}}$. So, as long as $v(x_{\underline{i}}) \neq 0$, we have $\lambda = w_{\underline{i}}^{2s}$.

Now suppose $v(x_{\underline{i}}) = 0$; then we may differentiate 3.5 and again evaluate at $z = x_{\underline{i}}$, giving

$$\begin{aligned}
|T_{\underline{i}}'(x_{\underline{i}})|^s T_{\underline{i}}'(x_{\underline{i}}) v'(T_{\underline{i}} x_{\underline{i}}) &= \lambda v'(x_{\underline{i}}) \\
w_{\underline{i}}^{2s} (-1)^{|\underline{i}|} w_{\underline{i}}^2 v'(x_{\underline{i}}) &= \lambda v'(x_{\underline{i}}).
\end{aligned}$$

So, as long as $v'(x_{\underline{i}}) \neq 0$, we have $\lambda = (-1)^{|\underline{i}|} w_{\underline{i}}^{2(s+1)}$. Continuing in this way, we may show that any eigenvalue of $\mathcal{L}_{\underline{i}}$ must be of the form $\lambda_k = (-1)^{k|\underline{i}|} w_{\underline{i}}^{2(s+k)}$ for some $k \in \mathbb{Z}^+$. In fact, every such λ_k is an eigenvalue of algebraic multiplicity one (see [May76]). Therefore, we may write

$$\operatorname{tr}(\mathcal{L}_{\underline{i}}) = \sum_{k=0}^{\infty} w_{\underline{i}}^{2s} \cdot \left((-1)^{|\underline{i}|} w_{\underline{i}}^2 \right)^k = \frac{w_{\underline{i}}^{2s}}{1 - (-1)^{|\underline{i}|} w_{\underline{i}}^s}.$$

Hence we have the following formula for the trace of $\mathcal{L}_s^n$:

$$\operatorname{tr}(\mathcal{L}_s^n) = \sum_{|\underline{i}|=n} \operatorname{tr}(\mathcal{L}_{\underline{i}}) = \sum_{|\underline{i}|=n} \frac{w_{\underline{i}}^{2s}}{1-(-1)^n w_{\underline{i}}^2}.$$

Substituting this into our formula for the Fredholm determinant (equation (3.2)), we have

$$\det(I - z\mathcal{L}_s) = 1 + \sum_{n=1}^{\infty} z^n d_n(s),$$

where

$$d_n(s) = \sum_{r=1}^{\infty} \sum_{n_1+\cdots+n_r=n} \frac{(-1)^r}{r!} \prod_{j=1}^{r} \frac{1}{n_j} \sum_{|\underline{i}|=n} \frac{w_{\underline{i}}^{2s}}{1-(-1)^n w_{\underline{i}}^2}.$$

Recall that, for the purposes of computation, we instead consider $D(s,z) = \det(I - z^2\mathcal{L}_s^2)$. In this case, we have

$$D(s,z) = 1 + \sum_{n=1}^{\infty} z^n b_n(s),$$

where

$$b_n(s) = \sum_{k=1}^{n-k} (-1)^k d_k(s) d_{n-k}(s).$$

Alternatively, we may note that $D(s,z)$ is given by

$$\begin{aligned}
D(s,z) &= \det(I - z\mathcal{L}_s)\det(I + z\mathcal{L}_s) \\
&= \exp \operatorname{tr}(\log(I - z\mathcal{L}_s) + \log(I + z\mathcal{L}_s)) \\
&= \exp \operatorname{tr}\left(\left(-\sum_{n=1}^{\infty} \frac{z^n}{n}\mathcal{L}_s^n\right) + \left(\sum_{n=1}^{\infty} \frac{z^n}{n}(-1)^{n+1}\mathcal{L}_s^n\right)\right) \\
&= \exp\left(-\sum_{n=1}^{\infty} \frac{z^n}{n} \operatorname{tr}(\mathcal{L}_s^n) + (-1)^n \operatorname{tr}(\mathcal{L}_s^n)\right) \\
&= \exp\left(-\sum_{n=1}^{\infty} z^n C_n(s)\right),
\end{aligned}$$

where

$$C_n(s) = \begin{cases} 0 & \text{if n is odd} \\ \frac{2}{n}\sum_{|\underline{i}|=n} \frac{w_{\underline{i}}^{2s}}{1-(-1)^n w_{\underline{i}}^2} & \text{if n is even.} \end{cases}$$

Hence, by the same working as for equation (3.2), we have

$$D(s,z) = 1 + \sum_{n=1}^{\infty} b_n(s) z^n,$$

where for all $k \in \mathbb{N}$,

$$\begin{aligned}
b_{2k+1}(s) &= 0; \\
b_{2k}(s) &= \sum_{r=1}^{\infty} \sum_{n_1+\cdots+n_r=k} \frac{(-1)^r}{r!} \prod_{j=1}^{r} \frac{2}{n_j} \sum_{|\underline{i}|=2n_j} \frac{w_{\underline{i}}^{2s}}{1-w_{\underline{i}}^2}.
\end{aligned}$$

This completes the proof of Theorem 3.1.3 (a).

3.3 Proof of (b)

The proof of Theorem 3.1.3 (b) consists of three parts. First, we show that the coefficients $d_n(s)$ in the power series expansion of $\det(I - z\mathcal{L}_s)$ are bounded by $K\theta^{n^2}$ for some $0 < \theta < 1$, $K > 0$. We then prove that this bound implies a bound of the same order on the coefficients $b_n(s)$ in the expansion of $D(s, z) = \det(I - z\mathcal{L}_s^2)$. Finally, we show that this bound results in a bound, also of the same order, on the error of truncations of the power series for $D(s, z)$, proving the result.

Step 1 - Bounding $\mathbf{d_n}$(s)

First, we have the following formal expression for the Fredholm determinant (see [Gro54]):

$$\det(I - z\mathcal{L}_s) = \begin{vmatrix} 1 - z\lambda_1\hat{l}_1(\hat{g}_1) & -z\lambda_1\hat{l}_1(\hat{g}_2) & -z\lambda_1\hat{l}_1(\hat{g}_3) & \cdots \\ -z\lambda_2\hat{l}_2(\hat{g}_1) & 1 - z\lambda_2\hat{l}_2(\hat{g}_2) & -z\lambda_2\hat{l}_2(\hat{g}_3) & \cdots \\ -z\lambda_3\hat{l}_3(\hat{g}_1) & -z\lambda_3\hat{l}_3(\hat{g}_2) & 1 - z\lambda_3\hat{l}_3(\hat{g}_3) & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{vmatrix},$$

which leads to the following representation of $\det(I - z\mathcal{L}_s)$:

Lemma 3.3.1. *We may write* $\det(I - z\mathcal{L}_s) = 1 + \sum_{n=1}^{\infty}$, *where*

$$d_n(s) = (-1)^n \sum_{i_1 < \cdots < i_n} \lambda_{i_1} \cdots \lambda_{i_n} \det[\hat{l}_{i_j}(\hat{g}_{i_k})]_{j,k=1}^n,$$

and $\det[\hat{l}_{i_j}(\hat{g}_{i_k})]_{j,k=1}^n$ *denotes the determinant of the* $n \times n$ *matrix with entries* $\hat{l}_{i_j}(\hat{g}_{i_k})$, $1 \leq j, k \leq n$, *where* $\hat{l}_{i_j}$ *and* $\hat{g}_{i_k}$ *are as defined in section 3.2.*

We aim to bound the power series coefficients $d_n(s)$ using this alternative representation. We have the following (see [Had93]):

Theorem 3.3.2 (Hadamard's Inequality)**.** *If A is an $n \times n$ matrix with entries of modulus at most 1, then* $|\det A| \leq n^{n/2}$.

Recall that $||\hat{l_n}|| = ||\hat{g_n}|| = 1$ by definition. Hence $|\hat{l}_{i_j}(\hat{g}_{i_k})| \leq 1$ for all $1 \leq j$, $k \leq n$. Therefore, by Hadamard's Inequality (Theorem 3.3.2), $|\det[\hat{l}_{i_j}(\hat{g}_{i_k})]_{j,k=1}^n| \leq n^{\frac{n}{2}}$. Hence

$$|d_n(s)| \leq \left(\sum_{i_1 < \cdots < i_n} \lambda_{i_1} \cdots \lambda_{i_n} \right) n^{\frac{n}{2}},$$

so it remains to prove a sufficient bound on $\sum_{i_1 < \cdots < i_n} \lambda_{i_1} \cdots \lambda_{i_n}$. From the proof of (a) (section 3.2), we have $\lambda_n \leq \frac{225}{44}\gamma^n$ for all $n \geq 1$, with $\gamma = \left(\frac{44}{45}\right)^{\frac{1}{2}}$.

We make use of the following result: for any $0 < \gamma < 1$, $k_1 \geq 0$, we have

$$\begin{aligned}
\sum_{k_1 < \cdots < k_n} \gamma^{k_1 + \cdots + k_n} &= \sum_{k_1=1}^{\infty} \left(\sum_{k_2=k_1+1}^{\infty} \left(\cdots \left(\sum_{k_n=k_{n-1}+1}^{\infty} \gamma^{k_1+\cdots+k_n} \right) \cdots \right) \right) \\
&= \sum_{k_1=1}^{\infty} \left(\cdots \left(\sum_{k_{n-1}=k_{n-2}+1}^{\infty} \gamma^{k_1+\cdots+k_{n-1}} \left(\sum_{k_n=k_{n-1}+1}^{\infty} \gamma^{k_n} \right) \right) \cdots \right) \\
&= \sum_{k_1=1}^{\infty} \left(\cdots \left(\sum_{k_{n-1}=k_{n-2}+1}^{\infty} \gamma^{k_1+\cdots+k_{n-1}} \left(\sum_{m=0}^{\infty} \gamma^{k_{n-1}+1+m} \right) \right) \cdots \right) \\
&= \sum_{k_1=1}^{\infty} \left(\cdots \left(\sum_{k_{n-1}=k_{n-2}+1}^{\infty} \gamma^{k_1+\cdots+2k_{n-1}} \left(\sum_{m=0}^{\infty} \gamma \cdot \gamma^{m} \right) \right) \cdots \right) \\
&= \sum_{k_1=1}^{\infty} \left(\cdots \left(\sum_{k_{n-1}=k_{n-2}+1}^{\infty} \gamma^{k_1+\cdots+2k_{n-1}} \right) \cdots \right) \frac{\gamma}{(1-\gamma)} \\
&\cdots \\
&= \sum_{k_1=1}^{\infty} \left(\cdots \left(\sum_{k_{n-2}=k_{n-3}+1}^{\infty} \gamma^{k_1+\cdots+3k_{n-2}} \right) \cdots \right) \frac{\gamma \cdot \gamma^2}{(1-\gamma)(1-\gamma^2)} \\
&\cdots \\
&= \frac{\gamma \cdots \gamma^n}{(1-\gamma)\cdots(1-\gamma^n)} = \frac{\gamma^{\frac{n(n+1)}{2}}}{(1-\gamma)\cdots(1-\gamma^n)}.
\end{aligned}$$

Hence, we have the following bound on the power series coefficients $d_n(s)$:

$$|d_n(s)| \leq C \left(\frac{225}{44} \right)^n \cdot \left(\frac{44}{45} \right)^{\frac{n(n+1)}{4}} \cdot n^{\frac{n}{2}},$$

with the constant $C = \prod_{r=1}^{\infty} \left(1 - \left(\frac{44}{45} \right)^{\frac{r}{2}} \right)^{-1}$.

It remains to show that for sufficiently large n[1], $|d_n(s)| \leq K\theta^{n^2}$ for some $0 < \theta < 1$, $K > 0$. First, fix $0 < D < \frac{1}{2} \log \frac{45}{44}$, and note that for sufficiently large n, we have that $\log n \leq D n^{\varepsilon}$ for all $0 < \varepsilon \leq 1$. Then we have

$$\begin{aligned}
\log |d_n(s)| &\leq \log C + n \log \left(\frac{225}{44} \right) + \frac{n(n+1)}{4} \log \left(\frac{44}{45} \right) + \frac{n}{2} \log n \\
&\leq \log C + n \log \left(\frac{225}{44} \right) + \frac{n(n+1)}{4} \log \left(\frac{44}{45} \right) + \frac{D}{2} n^2 \\
&= \log C + n \left(\log \left(\frac{225}{44} \right) + \frac{1}{4} \log \left(\frac{44}{45} \right) \right) + n^2 \left(\frac{1}{4} \log \left(\frac{44}{45} \right) + \frac{D}{2} \right) \\
&= Z + Yn + Xn^2 \leq Wn^2
\end{aligned}$$

[1]We write 'X is true for sufficiently large n' to mean 'there exists some N such that X is true for all $n > N$'.

for sufficiently large n and some $W < 0$ (since $X < 0$ by our bound on D). It follows that

$$|d_n(s)| \leq e^{Wn^2} = K\theta^{n^2}$$

with $0 < \theta = e^W < 1$, $K = 1 > 0$.

Step 2 - Bounding $\mathbf{b_n}$(s)

Recall that we define

$$b_n(s) = \sum_{n=1}^{k-1} d_k(s)d_{n-k}(s).$$

Hence, we can bound the modulus of $b_n(s)$ by

$$\begin{aligned}|b_n(s)| &\leq \sum_{k=1}^{n-1} |d_k(s)|d_{n-k}(s)| \\ &\leq \sum_{k=1}^{n-1} \left(K\theta^{k^2}\right)\left(K\theta^{(n-k)^2}\right) \\ &= \sum_{k=1}^{n-1} K\theta^{n^2-2nk+2k^2}.\end{aligned}$$

Let $f(k) = n^2 - 2nk + 2k^2$. Then $f'(k) = 4k - 2n$, so $f(k)$ achieves a minimum (and $\theta^{f(k)}$ achieves a maximum) at $k = \frac{n}{2}$. We then have $f(\frac{n}{2}) = n^2 - 2\frac{n^2}{2} + 2\left(\frac{n}{2}\right)^2 = \frac{n^2}{2}$, meaning we can write

$$|b_n(s)| \leq K\sum_{k=1}^{n-1} \theta^{\frac{n^2}{n}} = K(n-1)\theta^{\frac{n^2}{n}}.$$

Using a similar argument to step 1, we have

$$\log|b_n(s)| \leq Z' + X'n^2 \leq W'n^2$$

for some $W < 0$. Hence

$$|b_n(s)| \leq e^{Wn^2} = (\theta')^{n^2},$$

with $0 < \theta' < 1$.

Step 3 - Bounding the Error

It remains to show that this bound on the coefficients $b_n(s)$ implies a bound of the same order on the error when truncating the power series $D(s,1) = 1 + \sum_{n=1}^{\infty} b_n(s)$ at some $N \geq 1$. Recall that we write s_N to mean the largest solution $0 \leq s_N \leq 1$ to the equation $1 + \sum_{n=1}^{N} b_n(s) = 0$. We write $s = \dim_H E_{2,5}$, and note that this means s is the largest solution $0 \leq s \leq 1$ to the equation $D(s,1) = 0$. We aim to show that there exist $0 < \theta < 1$, $K > 0$ such that $|s_N - s| < K\theta^{N^2}$ for all $N \geq 1$.

We introduce the following notation:

$$F(t) = 1 + \sum_{n=1}^{\infty} b_n(s), \quad F_N(t) = 1 + \sum_{n=1}^{N} b_n(s).$$

Then, by the Mean Value Theorem, there exists t_N between s_N and s such that

$$(s_N - s)F'_N(t_N) = F_N(s_N) - F_N(s).$$

Since $F_N(s_N) = 0 = F(s)$, we may instead write

$$(s_N - s)F'_N(t_N) = F(s) - F_N(s) = \sum_{n=N+1}^{\infty} b_n(s).$$

Hence, we have the following:

$$\begin{aligned}
|s_N - s| &= \left| \frac{1}{F'_N(t_N)} \sum_{n=N+1}^{\infty} b_n(s) \right| \leq \frac{1}{|F'_N(t_N)|} \sum_{n=N+1}^{\infty} K\theta^{n^2} \\
&= \frac{1}{|F'_N(t_N)|} \sum_{r=1}^{\infty} K\theta^{(N+r)^2} = \frac{1}{|F'_N(t_N)|} \sum_{r=1}^{\infty} K\theta^{N^2+2Nr+r^2} \\
&= \frac{K\theta^{N^2}}{|F'_N(t_N)|} \sum_{r=1}^{\infty} \theta^{2Nr+r^2} \leq \frac{K\theta^{N^2+1}}{|F'_N(t_N)|} \sum_{r=1}^{\infty} \theta^{2Nr} \\
&= \frac{K\theta^{N^2+1}}{|F'_N(t_N)|} \cdot \frac{\theta^{2N}}{1-\theta^{2N}} \leq \frac{K\theta^{(N+1)^2}}{|F'_N(t_N)|(1-\theta)} \leq C\theta^{N^2}.
\end{aligned}$$

This completes the proof of Theorem 3.1.3 (b).

4 The Jenkinson-Pollicott Algorithm

The paper [JP01] implicitly describes an algorithm for estimating the dimension of sets of continued fractions whose coefficients are taken from a finite set of positive integers: the "Jenkinson-Pollicott Algorithm". Broadly, the algorithm finds the largest zero of the degree-N truncation of the power series for $D(s,1)$. That is, for a given N, it finds the unique s_N such that

$$F_N(s) = 1 + \sum_{n=1}^{N} b_n(s) = 0,$$

where the $b_n(s)$ are as defined in Theorem 3.1.3.

4.1 Overview

We give an outline of and motivation for the structure of the algorithm.

The terms $b_n(s)$ vanish when n is odd, so we focus on calculating the coefficients $b_{2k}(s)$ for a given $0 < s < 1$ and for all $1 \leq k \leq \frac{N}{2}$. We may write the formula for $b_{2k}(s)$ as follows

$$b_{2k}(s) = \sum_{r=1}^{\infty} \sum_{n_1+\cdots+n_r=k} \frac{(-1)^r}{r!} \prod_{j=1}^{r} C_{2n_j}(s),$$

where

$$C_{2n_j}(s) = \frac{1}{n_j} \sum_{|\underline{i}|=2n_j} \frac{w_{\underline{i}}^{2s}}{1-w_{\underline{i}}^{2}}.$$

Note that some elements of this formula depend only on k, while the terms $C_{n_j}(s)$ depend on s. Since we are looking for a zero of $F_N(s)$, we should expect to use a root-finding method which may evaluate the function F many times. As such, we separate out steps of the computation which do not depend on s, so that they need only be calculated once.

Hence, the structure of the algorithm is as follows:

1. For each $1 \leq k \leq \frac{N}{2}$, find the compositions of k;
2. For each $1 \leq n_j \leq N$, find the sequences of length n_j;
3. For each sequence $\underline{i}$, find the weight $w_{\underline{i}}$;
4. Use a numerical root-finding method to find the largest zero of $F_N(s)$, which is evaluated as follows:
    (i) For each $1 \leq n_j \leq \frac{N}{2}$, use the pre-computed weights to calculate $C_{2n_j}(s)$;
    (ii) For each $1 \leq k \leq \frac{N}{2}$, use the pre-computed compositions and values of $C_{2n_j}(s)$ to calculate $b_{2k}(s)$;
    (iii) Return 1+ the sum of the $b_{2k}(s)$.

4.2 Implementation

We give details of a Python implementation of the algorithm, using the `mpmath` package [MP23] to allow for calculations to arbitrary precision. Appendix A contains the full code.

Step 1 - Compositions

Recall that the *compositions* of a positive integer k are all <u>sequences</u> of positive integers that sum to k, while the *partitions* of k are all <u>multisets</u> of positive integers that sum to k. In other words, compositions care about order, while partitions do not. For any k, the two things that need to be computed before the function F can be evaluated are the compositions of k and the weights $w_{\underline{i}}$ of all sequences $\underline{i}$ of length $1 \leq n_j \leq 2k$.

To find the compositions of k, we first find the partitions and then permute them to get all the compositions. The algorithm uses Jerome Kelleher's Python implementation of his `AccelAsc` algorithm (see [KO14]), which finds all partitions of a given integer k, up to permutations. These are then permuted in all possible ways, using Python's `itertools` library, which gives us all the compositions. However, this list will contain duplicates, since for example the composition (2, 2, 1) of 5 may be permuted into itself by swapping the first two elements. So we need to remove all duplicate compositions from our list. Recall that we need to be able to access all compositions of k of length r, so we want to sort the compositions of k into groups of each length. Hence, the final step is to sort the consolidated compositions into lists of compositions of ascending length. The implementation of these last two steps is straightforward.

Step 2 - Sequences

The more complex task is finding the weights of all sequences of length $1 \leq n_j \leq 2k$ with entries in $\{2, 5\}$. Indeed, we must first find all such sequences of length n_j. This is achieved with a recursive algorithm: define

$$f(n) = \begin{cases} \emptyset & \text{if } n = 0 \\ \{2, 5\} & \text{if } n = 1 \\ \bigcup_{\underline{i} \in f(n-1)} \{\underline{i}2, \underline{i}5\} & \text{otherwise} \end{cases}$$

where $\underline{i}r$ denotes the sequence $\underline{i}$ with r appended to the end. Then, given any $n \in \mathbb{Z}^+$, $f(n)$ returns the set containing all sequences of length n. Recursive definitions such as these lend themselves particularly well to implementation in code. In practice, the function that returns all sequences of length n reads as an almost one-to-one translation of this.

```python
def find_all_sequences(family, n):
if n <= 0:
    return ()
if n == 1:
    return [tuple([x]) for x in family]
else:
    sequences = []
    remainder = find_all_sequences(family, n - 1)
    for seq in remainder:
        for x in family:
            r = list(seq)
            r.append(x)
            sequences.append(tuple(r))
    return sequences
```

Step 3 - Weights

Recall that

$$w_{\underline{i}} = \prod_{k=0}^{|\underline{i}|-1} T^k x_{\underline{i}} = \prod_{k=0}^{|\underline{i}|-1} \overline{[i_{k+1}, \cdots, i_n, i_1, \cdots, i_k]}.$$

There are some important things to note from this definition. First, notice that cyclically permuting $\underline{i}$ does not change the weight, since the product runs over all cyclic permutations of $\underline{i}$. So, if we have two sequences $\underline{i}$ and $\underline{j}$ such that the cyclic permutation of $\underline{j}$ by k positions (for some $0 \leq k \leq n-1$) is equal to $\underline{i}$, then $w_{\underline{i}} = w_{\underline{j}}$. This means that when computing weights, we need only consider sequences up to cyclic permutations.

The other observation which results in a significant computational benefit is that if $\underline{j} = \underline{i}^r$, that is, $\underline{j}$ is the r-fold concatenation of $\underline{i}$, for some $r \in \mathbb{N}$, then $w_{\underline{j}} = w_{\underline{i}}^r$. We can see this by noting that $x_{\underline{j}} = x_{\underline{i}}$, and that $|\underline{j}| = r|\underline{i}|$. Then we have

$$\begin{aligned} w_{\underline{j}} &= \prod_{k=0}^{|\underline{j}|-1} T^k x_{\underline{j}} = \prod_{k=0}^{r|\underline{i}|-1} T^k x_{\underline{i}} \\ &= \left(\prod_{k=0}^{|\underline{i}|-1} T^k x_{\underline{i}} \right)^r = w_{\underline{i}}^r. \end{aligned}$$

This means that we need only consider "atomic" sequences: ones that cannot be constructed by r-fold concatenation in this way. Combining this with the previous observation, we see intuitively that we need only consider atomic sequences up to cyclic permutations (since self-concatenation and cyclic permutations commute).

We formalise this in the following way: let $G_n = \langle \sigma \rangle$, where $\sigma(i_1 \cdots i_n) = i_n i_1 \cdots i_{n-1}$. That is, σ cyclically permutes the entries of $\underline{i}$ one place to the right. Then $G_n \cong Z_n$ consists of all cyclic permutations of sequences of length n. We then define a relation by $\underline{i} \sim_n \underline{j} \iff \tau(\underline{i}) = \underline{j}$ for some $\tau \in G_n$. It is straightforward to check that $\sim_n$ is an equivalence relation, so we may define the equivalence classes of sequences of length n under this relation: $[\underline{i}] = \{\tau(\underline{i}) \mid \tau \in G_n\}$.

Define the *atom* of $\underline{i}$, $a(\underline{i})$, to be the shortest sequence that can be self-concatenated to make $\underline{i}$. Define the *period* of a sequence $\underline{i}$, $m(\underline{i})$, to be the length of its atom, and define the *frequency* of $\underline{i}$, $r(\underline{i})$, to be $\frac{n}{m}$. Then a sequence of length n is *atomic* if any of the following three (equivalent) conditions holds:

(i) $a(\underline{i}) = \underline{i}$;

(ii) $m(\underline{i}) = n$;

(iii) $r(\underline{i}) = 1$.

For a concrete example, consider the sequence $\underline{i} = 2525$. Its atom is $\mathrm{a}(\underline{i}) = 25$, so its period and frequency are equal with $\mathrm{m}(\underline{i}) = \mathrm{r}(\underline{i}) = 2$. The equivalence class of $\underline{i}$ is $[\underline{i}] = \{2525, 5252\}$. Notice that the atoms of the elements of the equivalence class are themselves equivalent; this is true in general, and in fact we have the stronger result $[\mathrm{a}(\underline{i})] = \{\mathrm{a}(\underline{j}) \mid \underline{j} \sim \underline{i}\}$.

We use this result to our advantage to speed up computation significantly: we need only calculate in full one weight per atomic equivalence class. If we choose one representative sequence from each atomic class, we can calculate the weight of this sequence and then allocate the same weight to each sequence in that class. If the atomic sequence is not of length n, then we raise the calculated weight to the power of the frequency of the original sequence. This means that for each non-atomic sequence of length n, we need to store its period and its representative atomic sequence, which we can then use to recover the weights; see Table 1.

$\underline{i}$	Rep. atom	r($\underline{i}$)	$w_{\underline{i}}$
2222	2	4	$(w_2)^4$
2225	2225	1	w_{2225}
2252	2225	1	w_{2225}
2255	2255	1	w_{2255}
2522	2225	1	w_{2225}
2525	25	2	$(w_{25})^2$
2552	2255	1	w_{2255}
2555	2555	1	w_{2555}
5222	2225	1	w_{2225}
5225	2255	1	w_{2255}
5252	25	2	$(w_{25})^2$
5255	2555	1	w_{2555}
5522	2255	1	w_{2255}
5525	2555	1	w_{2555}
5552	2555	1	w_{2555}
5555	5	4	$(w_5)^4$

Table 1: The sequences of length 4, along with their metadata and derived weights. For each $\underline{i}$, a representative from [a($\underline{i}$)] is stored, as well as the frequency r($\underline{i}$). Once the weight of each of the representatives is computed, the weights $w_{\underline{i}}$ are derived from the representative weights and the frequencies.

Despite this effort, we do still need to compute some weights the usual way. Recall that the weight $w_{\underline{i}}$ of $\underline{i}$ is calculated by taking the product of all the periodic continued fractions with coefficients given by cyclic permutations of $\underline{i}$. So if $|\underline{i}| = n$, to calculate $w_{\underline{i}}$ we must compute n periodic continued fractions and multiply the results together. Computing the value of a periodic continued fraction turns out to be difficult; for a general n, there is no formula for the value of $[\overline{i_1, \cdots, i_n}]$. We may easily find formulae for small values of n using substitution, but as n increases the algebra quickly becomes unwieldy. One solution is to simply repeat the periodic part of the coefficients many times and treat this as a finite continued fraction, which may be evaluated using repeated division and addition.

$$[\overline{i_1, \cdots, i_n}] \approx [i_1, \cdots, i_n, i_1, \cdots, i_n, \cdots, i_1, \cdots, i_n]$$

In order to achieve a sufficiently accurate estimate, we take 200 copies of the sequence of coefficients. Computing this is very slow in practice, since in particular each continued fraction evaluation involves hundreds of division operations, which are slow for high-precision calculations; see section 4.3 for further analysis.

Step 4 - Root-Finding

We now want to find the largest zero $0 < s < 1$ of $F_N(s)$. To do so, we use `mpmath`'s built-in implementation of Ridders' method [Rid79]. This method was chosen as it doesn't need to be provided an initial value close to the root, instead it takes an interval on which the root lies; naturally we use the interval (0, 1).

At each iteration, the root-finding algorithm evaluates the function F_N twice. Evaluation of F_N is a straightforward implementation of the formulae for it and for the terms $b_{2k}(s)$ and $C_{2n_j}(s)$, using the compositions and weights computed in steps 1 and 3. The only "trick" employed is to compute and store all of the terms $C_{2n_j}(s)$ for $1 \leq n_j \leq \frac{N}{2}$ before computing the power series coefficients $b_{2k}(s)$, since each $C_{2n_j}(s)$ may appear multiple times in each $b_{2k}(s)$, and in multiple terms; for example:

$$\begin{aligned} b_4(s) &= -C_4(s) + \frac{1}{2}C_2(s)^2 \\ b_6(s) &= -C_6(s) + C_2(s)c_4(s) - \frac{1}{6}C_2(s)^3. \end{aligned}$$

4.3 Results & Analysis

We present calculations for the dimension of $E_{2,5}$ at different truncation levels N, as well as estimates of the dimensions of other sets E_X with X a finite collection of positive integers. Note that our estimate for $\dim E_2$ agrees with Jenkinson & Pollicott's up to the 25 decimal places given in [JP01].

N	s_N
2	0.289 437 334 307 471 705 153 754 589 692 696 429 906 938 365 855 316 6
4	0.285 962 548 449 455 696 072 238 249 372 286 252 023 552 765 656 753 6
6	0.285 964 050 706 904 893 556 878 216 077 242 057 307 730 271 999 792 2
8	0.285 964 050 689 145 602 358 503 834 800 911 629 186 692 920 593 049 3
10	0.285 964 050 689 145 608 345 949 039 125 200 084 296 483 279 832 013 5
12	0.285 964 050 689 145 608 345 948 980 972 327 436 157 142 957 067 963 6
14	0.285 964 050 689 145 608 345 948 980 972 327 452 519 963 101 709 553 3
16	0.285 964 050 689 145 608 345 948 980 972 327 452 519 963 101 575 698 1

Table 2: Estimates s_N of the dimension of $E_{2,5}$ for increasing values of N.

The estimates in table 3 are given to 25 decimal places, and were calculated using varying degrees of truncation; when X has more entries, the computations take longer, so we used smaller values of N for larger sets X.

X	Estimate of $\dim E_X$
$\{1, 2\}$	0.531 280 506 277 205 141 624 468 6
$\{1, 2, 3, 4, 5\}$	0.836 829 789 428 015 590 140 026 9
$\{1, 3, 5, 7, 9\}$	0.675 249 486 924 737 797 449 733 2
$\{2, 4, 6, 8\}$	0.480 113 128 214 559 687 018 892 3
$\{1, 10, 100, 1000\}$	0.347 848 057 508 714 199 889 760 0
$\{2001, 2018, 2023\}$	0.072 202 396 787 093 968 348 434 5

Table 3: Estimates of the dimension of E_X for various sets X.

Some informal discussion of algorithmic time complexity follows, focusing on calculating weights. Assuming there are two integers in our collection X, there are 2^n sequences of length n with entries in X. A calculation of s_N involves calculating weights for each of the 2^n sequences of length n, for all even n with $2 \leq n \leq N$. So we expect to calculate all the weights needed to compute s_N in $O(2^N)$ time.

Compare this with a straightforward analysis of the program's execution, which reveals that in fact the most time-consuming part of the algorithm in practice is calculating the weights, by a wide margin. This is due to the method used to estimate the values of the periodic continued fractions, which involves many division operations on high-precision numbers. Figure 4 shows the execution time for each estimate in table 2; note that the vertical axis is a log-scale, so we can see that the execution time is roughly exponential in N, however it appears to be faster than exponential for larger values of N. This is because of the speedup from our work on atomic sequences and cyclic permutation classes, which means that although there are 2^n sequences of length n, we don't need to calculate them all in full to have all the weights of sequences of length n.

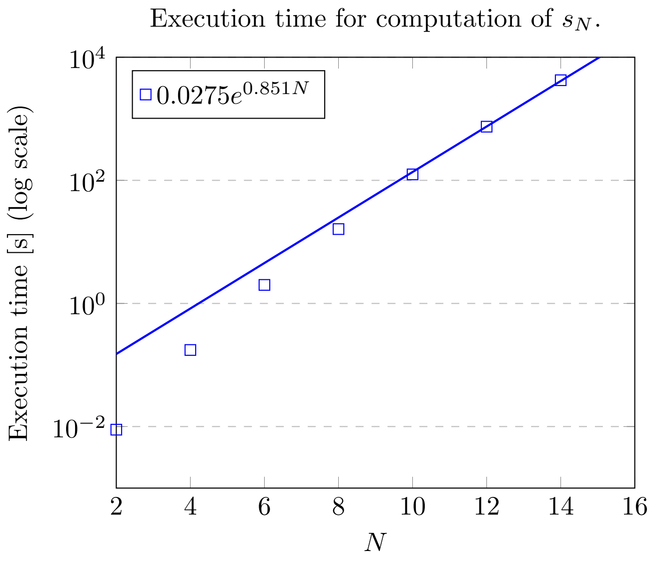

Figure 4: Execution time for different estimates s_N of $\dim E_{2,5}$, with a trendline.

We may wonder what effect the other parts of the algorithm have on the runtime of the program. First we find the compositions of m and sequences of length n for all $1 \leq m \leq \frac{N}{2}$ and $2 \leq n \leq N$ even. There are 2^{m-1} compositions of m and 2^n sequences of length n, however neither of the algorithms to find these involves any intensive operations (such as the division which makes calculating the weights so slow) so although they are still $O(2^n)$, the constant factor is much smaller than for the (less than) $O(2^n)$ complexity of calculating the weights. Hence, the weight calculation outweighs[2] the other computations. This example illustrates one of the ways in which big-O analysis can be misleading when it comes to practical applications. Further, since we determined that the weight calculation is actually slightly faster than $O(2^n)$, we should expect that for very large values of N, the steps to find the compositions and sequences will actually begin to outweigh the weight calculations. However, actually running the program for values of N large enough for this to take effect would likely take longer than the age of the universe, so this isn't of much concern. So although there are some obvious improvements we could make to our composition- and sequence-finding algorithms (e.g. explicitly using the list of sequences of length $n-1$ to find the sequences of length n, or taking a more direct approach to finding compositions rather than going via permutations), we needn't worry about these in practice.

Another general concern of algorithmic complexity is the memory usage, or *space complexity*, of a program; that is, how much space is used to store data during execution. Often, one can reduce

[2]Pun not intended.

execution time by increasing memory usage, for example by storing the results of computations so that they only need to be calculated once. This was exactly our approach in the weight-calculation section of the algorithm, wherein we reduced the number of weights to be computed by storing metadata (i.e. atoms and periods) about the sequences. This allowed us to build a full set of weights by only computing a subset of weights in full, at the cost of increased memory use at runtime. The amount data stored isn't significant, so this increased memory usage isn't an issue – at least for the relateively-small values of N for which we are running the program. It's important to note that these kinds of tradeoffs aren't always clear-cut, as there is some computational overhead introduced in storing, retrieving, and manipulating data. In our case, these overheads are much less than the time saved, so we happily take this tradeoff.

A Code

This appendix contains Python code for our implementation of the Jenkinson-Pollicott algorithm, as detailed in section 4. This code was written and tested with Python 3.11.4, using the `mpmath` package, version 1.2.1-2. Note that most of the arithmetic operations are performed using the `mpmath` package; the result of this is that these operations are written in function notation as opposed to the standard infix notation, making the code somewhat less readable for those not used to this notation. The function names are mostly self-explanatory, but we provide a reference below nonetheless (the `f` in these function names stands for "floating point").

$$\begin{aligned}
\texttt{mp.mpf(x)} &= x \\
\texttt{mp.fadd(x, y)} &= x + y \\
\texttt{mp.fmul(x, y)} &= x \times y \\
\texttt{mp.fdiv(x, y)} &= x \div y \\
\texttt{mp.power(x, y)} &= x^y \\
\texttt{mp.factorial(x)} &= x!
\end{aligned}$$

```python
import itertools
from mpmath import mp
import time


class SequenceLookup:
    '''
    Class to store information about sequences in order to find all weights
    without computing them all.
    '''
    def __init__(self, atom, freq):
        self.atom = atom
        self.freq = freq
        self.weight = None

    def setAtom(self, a):
        self.atom = a

    def setWeight(self, w):
        self.weight = w

    def isAtomic(self):
        if self.freq == 1:
            return True
        else:
            return False


def rotate_list(lst, n):
    '''
    Returns a list with the same entries as lst, cycled n places to the left.
    '''
    return lst[n:] + lst[:n]


def find_all_sequences(family, n):
    '''
    Returns all sequences of length n with entries in the list family.
    '''
    if n <= 0:
        return ()
    if n == 1:
        return [tuple([x]) for x in family]
    else:
        seq = []
        remainder = find_all_sequences(family, n - 1)
        for s in remainder:
            for x in family:
                r = list(s)
                r.append(x)
                seq.append(tuple(r))
        return seq


def weight(s, acc):
    '''
    Returns the weight of the sequence s.
    '''
    seq = list(s)
    w = mp.mpf(1)

    # Cycle seq, find the value of the purely periodic cf whose coefficients
    # are the cycled seq, and multiply w by this value. Repeat for each cycle.
    for i in range(len(seq)):
        w = mp.fmul(w, precise_cf_evaluate(rotate_list(seq, i) * acc))
    return w


def precise_cf_evaluate(cont_frac):
    '''
    Returns the real number with continued fraction coefficients given by the
    list cont_frac. Adapted from the cf_evalute function in the contfrac
    library in order to support mpmath operations. Also assumes the leading
    coefficient is a zero and has been omitted, i.e. the value is between 0
    and 1 (as is always the case for us).
    '''
    if len(cont_frac) == 0:
        return 0

    fraction = 0
    for coefficient in reversed(cont_frac):
        fraction = mp.fdiv(1, (mp.fadd(coefficient, fraction)))
    return fraction


def get_cyclic_perm_classes(seq, n):
    '''
    Returns the cyclic permutation classes for the sequences of length n in the
    list seq.
    '''
    cyclic_perm_classes = []

    for s in seq:
        break_flag = False
        for c in cyclic_perm_classes:
            for i in range(n):
                r = rotate_list(s, i)
                if r == c[0]:
                    c.append(s)
                    break_flag = True
                    break

            if break_flag:
                break

        if break_flag:
            continue

        cyclic_perm_classes.append([s])

    return cyclic_perm_classes


def initialise_lookup(family, n):
    '''
    Returns a list of SequenceLookup objects, one for each sequence of length
    n. Each SequenceLookup has its atom and freq fields correctly assigned,
    but no weight assigned, as this is handled by the function get_full_lookup.
    '''
    seq = find_all_sequences(family, n)

    # For each sequence of length n, find its atom and create a SequenceLookup
    # object with the atom and the frequency of the sequence.
    # A potential speedup here would be to break when i > n/2, since the
    # frequency must divide n.
    seq_atoms = []
    for s in seq:
        for i in range(1, n + 1):
            r = rotate_list(s, i)
            # We cycle the sequence until the cycled version is equal to the
            # uncycled version; the sequence is atomic iff we cycle all the way
            # through the sequence, else we know the atomic part and create a
            # SequenceLookup object with the relevant information.
            if s == r:
                atom = s[:i]
                freq = n // i
                seq_atoms.append(SequenceLookup(atom, freq))
                break

    # Create a dictionary which assigns SequenceLookup objects to their
    # original sequences.
    seq_lookup = dict(zip(seq, seq_atoms))

    # Assign the same atom to all sequences in the same cyclic permutation
    # class.
    cyclic_perm_classes = get_cyclic_perm_classes(seq, n)
    for s in seq_lookup:
        for c in cyclic_perm_classes:
            if s in c:
                period = n // seq_lookup[s].freq
                seq_lookup[s].setAtom(c[0][:period])
                break

    return seq_lookup
```

```
def get_full_lookup(family, N, acc):
    '''
    Returns a list of SequenceLookup objects, one for each sequence of length
    m with entries in family, with 1 <= m <= n. Each SequenceLookup has all of
    its fields correctly assigned.
    '''
    lookup =  [initialise_lookup(family, n) for n in range(N + 1)]

    # Assign weights to elements of lookup by the following logic:
    # if the sequence is atomic, calculate its weight from scratch;
    # else, use its freq and the weight of its atom to find its weight.
    for l in lookup:
        for s in l:
            if l[s].isAtomic():
                w = weight(l[s].atom, acc)
            else:
                atm = l[s].atom
                frq = l[s].freq
                n = len(atm)
                w = mp.power(lookup[n][atm].weight, frq)

            l[s].setWeight(w)

    return lookup


def accel_asc(n):
    '''
    Returns a list containing all partitions of n.
    Code from https://jeromekelleher.net/generating-integer-partitions.html
    '''
    a = [0 for i in range(n + 1)]
    k = 1
    y = n - 1
    while k != 0:
        x = a[k - 1] + 1
        k -= 1
        while 2 * x <= y:
            a[k] = x
            y -= x
            k += 1
        l = k + 1
        while x <= y:
            a[k] = x
            a[l] = y
            yield a[:k + 2]
            x += 1
            y -= 1
        a[k] = x + y
        y = x + y - 1
        yield a[:k + 1]


def find_all_compositions(k):
    '''
    Returns a list containing k lists of lists of integers. The n-th list
    consists of all compositions of k of length n.
    '''
    if k == 0:
        return None

    # Find all partitions of k and sort them by length.
    partitions = list(accel_asc(k))
    partitions.sort(key=lambda p:len(p))

    # Permute all partitions to get all compositions.
    compositions = []
    for p in partitions:
        compositions.append(list(itertools.permutations(p)))

    # Remove duplicate compositions (weird Python way).
    consolidated_compositions = []
    for c in compositions:
        consolidated_compositions.append(list(dict.fromkeys(c)))

    # Group partitions of the same length into lists.
    grouped_partitions = [[] for i in range(k)]
    i = 0
    for c in consolidated_compositions:
        for q in c:
            if len(q) != i+1:
                i += 1
            grouped_partitions[i].append(list(q))

    return grouped_partitions


def C(s, n, weights):
    '''
    Returns C_n(s) for a given n and s.
    '''
    if n % 2 == 1 or n <= 0:
        return 0

    sum = mp.mpf(0)
    for w in weights:
        frac = mp.fdiv(mp.power(w, mp.fmul(2 ,s)), (1 - mp.power(w, 2)))
        sum = mp.fadd(sum, frac)
    return mp.fmul(sum, mp.fdiv(2, n))


def b(k, compositions, C_n):
    '''
    Returns b_2k(s) for a given k and precomputed partitions and C_n(s).
    '''
    if k <= 0:
        return 0

    sum = mp.mpf(0)
    for j in range(k):
        for p in compositions[k][j]:
            r = len(p)
            factor = mp.fdiv(mp.power(-1, r), mp.factorial(r))
            prod = mp.mpf(1)
            for n in p:
                prod = mp.fmul(prod, C_n[2*n])
            sum = mp.fadd(sum, (mp.fmul(factor, prod)))
    return sum


# Set the decimal precision for the mpmath package.
mp.dps = 52

# The subset of the natural numbers which we restict our coefficients to.
family = [mp.mpf(2), mp.mpf(5)]

# Accuracy parameter - this controls how accurate our continued fraction
# calculations are. Higher is more accurate but slower. A value of 200 gives
# no error (compared to JP2001) when family = [1, 2], so we take this to be a
# safe value for all sets.
acc = 200


print("family =", family)
print("acc =", acc)

for N in range(2, 17, 2):
    time_start = time.time()

    lookup = get_full_lookup(family, N, acc)

    compositions = [find_all_compositions(k) for k in range(0, (N // 2) + 1)]

    weights = [[l[s].weight for s in l] for l in lookup]

    def est(s, N=N, compositions=compositions, weights=weights):
        '''
        Returns an estimate of the function D(s, 1). In particular, it is the
        degree-N truncation of the sum.
        '''
        C_n =  [C(s, n, weights[n]) for n in range(N + 1)]
        sum = mp.mpf(1)
        for k in range(1, (N // 2) + 1):
            sum = mp.fadd(sum, b(k, compositions, C_n))
        return sum

    print("N =", N)

    # mpmath function to find and print the root of est in the interval (0, 1).
    mp.nprint(mp.findroot(est, (0, 1), solver="ridder", verbose=False), mp.dps)

    time_end = time.time()

    print("time:", time_end - time_start, "seconds")
```

References

[Ahl79] L. Ahlfors. *Complex Analysis: An Introduction to The Theory of Analytic Functions of One Complex Variable.* McGraw-Hill Education, 1979.

[Bab78] K. I. Babenko. On a problem of Gauss. *Dokl. Akad. Nauk SSSR*, 238(5):1021–1024, 1978.

[Bol90] B. Bollobàs. *Linear Analysis : An Introductory Course.* Cambridge Mathematical Textbooks. Cambridge University Press, 1990.

[Bry85] V. Bryant. *Metric Spaces: Iteration and Application.* Cambridge University Press, 1985.

[Bum85] R. T. Bumby. Hausdorff dimension of sets arising in number theory. *Number Theory (New York, 1983–84)*, pages 1–8, 1985.

[BY78] K. I. Babenko and S. P. Yur'ev. On the discretization of a problem of Gauss. *Dokl. Akad. Nauk SSSR*, 240(6):1273–1276, 1978.

[Cus77] T. W. Cusick. Continuants with bounded digits. *Mathematika*, 24(2):166–172, 1977. doi:10.1112/S0025579300009050.

[Fal97] K. Falconer. *Techniques in Fractal Geometry.* Wiley, 1997.

[Fal03] K. Falconer. *Fractal Geometry: Mathematical Foundations and Applications.* Wiley, 2nd edition, 2003.

[FN17] R. S. Falk and R. D. Nussbaum. C^m Eigenfunctions of Perron-Frobenius Operators and a New Approach to Numerical Computation of Hausdorff Dimension, 2017, 1612.00870.

[Good41] I. Good. The fractional dimensional theory of continued fractions. *Mathematical Proceedings of the Cambridge Philosophical Society*, 37(3):199–228, 1941. doi:10.1017/S030500410002171X.

[Gro54] A. Grothendieck. Produits tensoriels topologiques et espaces nucléaires. In *Séminaire Bourbaki : années 1951/52 - 1952/53 - 1953/54, exposés 50-100*, number 2 in Séminaire Bourbaki. Société mathématique de France, 1954.

[Had93] J. S. Hadamard. Résolution d'une question relative aux determinants. *Bulletin des Sciences Mathematiques*, 17:240–246, 1893.

[Hen88] D. Hensley. The hausdorff dimensions of some continued fraction cantor sets. *Journal of Number Theory*, 33(2):182–198, 1988. doi:10.1016/0022-314X(89)90005-X.

[Hen96] D. Hensley. A polynomial time algorithm for the hausdorff dimension of continued fraction cantor sets. *Journal of Number Theory*, 58(1):9–45, 1996. doi:10.1006/jnth.1996.0058.

[Jar28] V. Jarník. Zur metrischen theorie der diophantischen approximationen. *Prace Matematyczno-Fizyczne*, 36(1):91–106, 1928.

[JP01] O. Jenkinson and M. Pollicott. Computing the dimension of dynamically defined sets: E_2 and bounded continued fractions. *Ergodic Theory and Dynamical Systems*, 21(5):1429–1445, 2001. doi:10.1017/S0143385701001687.

[JP18] O. Jenkinson and M. Pollicott. Rigorous effective bounds on the Hausdorff dimension of continued fraction Cantor sets: A hundred decimal digits for the dimension of E2. *Advances in Mathematics*, 325:87–115, 2018. doi:10.1016/j.aim.2017.11.028.

[Khi97] A. Khinchin. *Continued Fractions.* Dover Books on Mathematics. Dover Publications, 1997.

[KO14] J. Kelleher and B. O'Sullivan. Generating all partitions: A comparison of two encodings, 2014, 0909.2331.

[May76] D. H. Mayer. On a ζ function related to the continued fraction transformation. *Bulletin de la Société Mathématique de France*, 104:195–203, 1976. doi:10.24033/bsmf.1825.

[May91] D. Mayer. Continued fractions and related transformations. In T. Bedford, M. Keane, and C. Series, editors, *Ergodic theory, symbolic dynamics, and hyperbolic spaces*, chapter 7, pages 175–222. Oxford University Press, 1991.

[MP23] The mpmath development team. *mpmath: a Python library for arbitrary-precision floating-point arithmetic (version 1.3.0).* URL `http://mpmath.org/`. Accessed 2023-09-12.

[PP90] W. Parry and M. Pollicott. *Zeta Functions and the Periodic Orbit Structure of Hyperbolic Dynamics.* Asterisque; 187-188. Société mathématique de France, 1990.

[Rid79] C. Ridders. A new algorithm for computing a single root of a real continuous function. *IEEE Transactions on Circuits and Systems*, 26(11):979–980, 1979. doi:10.1109/TCS.1979.1084580.