ON THE SECOND LYAPUNOV EXPONENT OF SOME MULTIDIMENSIONAL CONTINUED FRACTION ALGORITHMS

VALÉRIE BERTHÉ, WOLFGANG STEINER, AND JÖRG M. THUSWALDNER

Abstract. We study the strong convergence of certain multidimensional continued fraction algorithms. In particular, in the two-dimensional case, we prove that the second Lyapunov exponent of Selmer’s algorithm is negative and bound away from zero. Moreover, we give heuristic results on several other continued fraction algorithms. Our results indicate that all classical multidimensional continued fraction algorithms cease to be strongly convergent for high dimensions. The only exception seems to be the Arnoux–Rauzy algorithm which, however, is defined only on a set of measure zero.

1. Introduction

In the present paper we study strong convergence properties of multidimensional continued fraction algorithms. In particular, we give results and empirical studies for the second Lyapunov exponent of such algorithms. One of our main objects is Selmer’s algorithm, which attracted a lot of interest in the recent years, in relation to an (unordered) continued fraction algorithm defined by Cassaigne in 2015. This algorithm, now called Cassaigne algorithm, was studied in the context of word combinatorics by Cassaigne, Labbé and Leroy in [CLL17] where it was shown to be conjugate to Selmer’s algorithm. Other properties of Selmer’s algorithm have been studied in [AL18, BFK15, BFK19, FS19, Sch01b, Sch04].

The first results on the second Lyapunov exponent of Selmer’s algorithm are due to Schweiger [Sch01b, Sch04], in terms of strong convergence holding almost everywhere, and to Nakaishi [Nak06], who proved that its second Lyapunov exponent \( \lambda_2(A_S) \) satisfies \( \lambda_2(A_S) < 0 \) for \( d = 2 \). This was conjectured already by Baldwin [Bal92a] (where Selmer’s algorithm is called generalized mediant algorithm, GMA for short, see also [Bal92b]); in particular, \( e^{\lambda_2(A_S)} \) is numerically calculated in [Bal92a, Table I on p. 1522]). Labbé [Lab15] heuristically calculated the Lyapunov exponents for the Cassaigne and Selmer algorithms (for \( d = 2 \)); this is actually the equality of these values that indicated the conjugacy of the algorithms. We mention that Bruin, Fokkink, and Kraaikamp [BFK15] give a thorough study of Selmer’s algorithm for \( d \geq 2 \); however, their proof of the fact that \( \lambda_2(A_S) < 0 \) is incomplete [BFK19]. The simplicity of the Lyapunov spectrum of the Cassaigne algorithm is proved by Fougeron and Skripchenko [FS19]. Heuristic calculations

Date: October 28, 2019.

This work was supported by the Agence Nationale de la Recherche through the project Codys (ANR-18-CE40-0007). The third author was supported by the projects FWF P27050 and FWF P29910 granted by the Austrian Science Fund. Part of this work has been done while the three authors were visiting the Erwin Schrödinger Institute in Vienna.
The second Lyapunov exponent of other algorithms are also provided by Baladi and Nogueira [BN96], see also [Nak02].

The proof of the negativity of the second Lyapunov exponent of Selmer’s algorithm provided by Nakaiishi [Nak06] is intricate. Moreover, Nakaiishi just deals with the case \( d = 2 \). This is also the case of Schweiger’s proof for almost everywhere strong convergence [Sch01b]. In the present paper we provide a simple proof for the fact that \( \lambda_2(A_S) < 0 \) which is based on ideas going back to Lagarias [Lag93] and Hardcastle and Khanin [Har02, HK02]. Moreover, we show that the involved matrices are Pisot whenever they are primitive, and we give a strictly negative upper bound for \( \lambda_2(A_S) \). For higher dimensions we provide heuristic results. These results indicate that Selmer’s algorithm is no longer strongly convergent for dimensions \( d \geq 4 \). For \( d = 3 \), the results of our computer programs (which are still running) do not allow yet to conclude that the second Lyapunov exponent is negative.

Another aim of this paper is to provide numerical calculations in order to obtain heuristic estimates for the second Lyapunov exponent of other well-known continued fraction algorithms. In particular, we consider the Brun algorithm, the Jacobi–Perron algorithm, the triangle map, and a new algorithm which is “in between” the Arnoux–Rauzy algorithm and Brun’s algorithm. It is interesting to see that apart from the Arnoux–Rauzy algorithm, which is strongly convergent in each dimension \( d \geq 2 \) (see [AD15]), all the other algorithms are no longer strongly convergent for high dimensions. Since the Arnoux–Rauzy algorithm is defined only on a set of zero measure (the so-called Rauzy gasket, see [AS13, AHS16]), we are not aware of any Markovian multi-dimensional continued fraction algorithm such as defined in Section 2 which acts on a set of positive measure and is strongly convergent in all dimensions. It was widely expected that the uniform approximation exponent, when it can be expressed in terms of the first and second Lyapunov exponents of the algorithm as \( 1 - \frac{\lambda_2}{\lambda_1} \) (see [Lag93, Theorem 1]) would be larger than 1 (and strictly smaller than Dirichlet’s bound \( 1 + \frac{1}{d} \)) for all \( d \geq 2 \); see e.g. [Lag93]. Our experimental studies tend to invalidate this conjecture.

Let us sketch the contents of this paper. The formalism of multidimensional continued fraction algorithms considered here is recalled in Section 2 together with the conditions given by Lagarias [Lag93]. The second Lyapunov exponent is mentioned in Section 3. We discuss the connections with the Paley–Ursell inequality in Section 4. We consider the Selmer algorithm in Section 5, the Brun algorithm in Section 6, the Jacobi–Perron algorithm in Section 7, a new algorithm inspired by the Arnoux–Rauzy algorithm in Section 8, and the triangle map in Section 9. Comparisons are provided in Section 10. Acknowledgment. We warmly thank Sébastien Labbé for his help with numerical simulations.

2. MULTIDIMENSIONAL CONTINUED FRACTION ALGORITHMS

We first introduce the formalism of multidimensional continued fraction algorithms that will be used in the following. Observe that algorithms mainly act here on sets of ordered entries (except for the Jacobi–Perron algorithm considered in Section 7). A \( d \)-dimensional algorithm acts on a set of dimension \( d \) for its renormalized version and of dimension \( d + 1 \).
for its homogeneous version. More precisely, for given \(d \geq 2\), let

\[
\Lambda = \{(x_0, x_1, \ldots, x_d) \in \mathbb{R}^{d+1} : x_0 \geq x_1 \geq \cdots \geq x_d \geq 0\},
\]

\[
\Delta = \{(x_1, \ldots, x_d) \in \mathbb{R}^d : 1 \geq x_1 \geq \cdots \geq x_d \geq 0\},
\]

\[\iota : \mathbb{R}^d \to \mathbb{R}^{d+1}, \quad (x_1, \ldots, x_d) \mapsto (1, x_1, \ldots, x_d)\].

In this paper, an (ordered) multidimensional continued fraction algorithm is a map

\[A : \Delta \to \text{GL}(d + 1, \mathbb{Z}) \quad \text{with} \quad \iota(x) A(x)^{-1} \in \Lambda \quad \text{for all} \quad x \in \Delta,
\]

together with the associated transformation

\[T : \Delta \to \Delta \quad \text{defined by} \quad \iota(Tx) \in \mathbb{R} \iota(x) A(x)^{-1}.
\]

Sometimes we use the associated linear, also called homogeneous or matrix version of \(T\) which is defined by

\[L : \Lambda \to \Lambda, \quad (x_0, x_1, \ldots, x_d) \mapsto (x_0, x_1, \ldots, x_d) A \left( \frac{x_1}{x_0}, \ldots, \frac{x_d}{x_0} \right)^{-1}.
\]

Then

\[A^{(n)}(x) = A(T^{n-1}x) \cdots A(Tx) A(x)
\]

is the cocycle associated with \(A\). It produces the \(d + 1\) sequences of rational convergents that are aimed to converge to \(x\). Indeed, writing

\[A^{(n)}(x) = \begin{pmatrix} q_0^{(n)} & p_{0,1}^{(n)} & \cdots & p_{0,d}^{(n)} \\ q_1^{(n)} & p_{1,1}^{(n)} & \cdots & p_{1,d}^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ q_d^{(n)} & p_{d,1}^{(n)} & \cdots & p_{d,d}^{(n)} \end{pmatrix},
\]

\(p_i^{(n)} = (p_{i,1}^{(n)}, \ldots, p_{i,d}^{(n)})\), we consider the convergence of \(\lim_{n \to \infty} p_i^{(n)}/q_i^{(n)}\) to \(x\), \(0 \leq i \leq d\).

The convergence is said to be weak if \(\lim_{n \to \infty} p_i^{(n)}/q_i^{(n)} = x\) for all \(i\) with \(0 \leq i \leq d\), and strong if \(\lim_{n \to \infty} |p_i^{(n)} - q_i^{(n)} x| = 0\) for all \(0 \leq i \leq d\).

Since we focus on the action of the matrices produced by the algorithm on the orthogonal space of \(\iota(x)\), we use left-multiplication for the description of the linear action in order to simplify notation and to avoid the use of the transpose.

Throughout this paper we suppose that a multidimensional continued fraction algorithm satisfies the following conditions which go back to Lagarias [Lag93]. Similar to [FS19] we just explain them briefly and refer to Lagarias’ paper for details.

(H1) **Ergodicity:** The map \(T\) admits an ergodic invariant probability measure \(\mu\) that is absolutely continuous with respect to Lebesgue measure.

(H2) **Covering Property:** The map \(T\) is piecewise continuous with non-vanishing Jacobian almost everywhere.

(H3) **Semi-weak convergence:** This is a mixing condition for \(T\) which implies weak convergence. For Markovian algorithms it can be checked by making sure
that the cylinders of the Markov partitions decrease geometrically. For some examples this is worked out in [Lag93]. See [FS19] for a sufficient condition expressed in terms of the existence of a special acceleration providing a simplex on which the induced algorithm is uniformly expanding.

(H4) Boundedness: This is log-integrability of the cocycle $A$ which is necessary in order to apply the Oseledets Theorem, i.e., the expectation of $\log(\max(||A||, 1))$ is finite.

(H5) Partial quotient mixing: This condition says that the expectation of the number $n$ for which $A^{(n)}(x)$ becomes a strictly positive matrix is finite.

Here and in the following, the Lyapunov exponents of the cocycle $A$ are denoted as

$$\lambda_1(A) \geq \lambda_2(A) \geq \cdots \geq \lambda_{d+1}(A).$$

Our motivation for studying the second Lyapunov exponent is due to the following result; see [HK00, Theorem 1] for a variant of this result and [Bal92a, Proposition 4].

**Proposition 2.1** ([Lag93, Theorem 4.1]). Let $\eta^*_A$ be the uniform approximation exponent of a $d$-dimensional multidimensional continued fraction algorithm $A$ satisfying conditions (H1) to (H5). We have $\lambda_1(A) > \lambda_2(A)$ and

$$\eta_A^*(x) = 1 - \frac{\lambda_2(A)}{\lambda_1(A)}$$

holds for almost all $x \in \Delta$. In particular, if $\lambda_2(A) < 0$ then $A$ is strongly convergent a.e.

We wish to show that $\lambda_2(A) < 0$ for various classical multidimensional continued fraction algorithms $A$.

### 3. The second Lyapunov exponent

The action of a continued fraction algorithm is given by a matrix $A$ acting by left-multiplication on some direction. To understand the quality of approximation, it is useful to work on the orthogonal of this direction. The action on the orthogonal is then given by the matrix $A$ acting by right-multiplication. We are thus interested in the action of the matrix $A$ and of the associated cocycle on a restricted hyperplane. By choosing a suitable basis of this hyperplane, the action of the algorithm is then described as a matrix that involves the usual differences that have the form $q_n x - p_n$ in the one-dimensional case, where $p_n/q_n$ are the convergents of $x$.

In order to give estimates of the second Lyapunov exponent of a multidimensional continued fraction algorithm we follow the ideas of Hardcastle and Khanin [Har02, HK02] who build on the work of Lagarias [Lag93].

Since $T$ is ergodic by (H1), the Lyapunov exponents of $A$ are the same for almost all $x \in \Delta$ w.r.t. the invariant measure of $T$. Under the conditions of Proposition 2.1, the Oseledets theorem gives, for generic $x \in \Delta$,

$$\lim_{n \to \infty} \frac{1}{n} \log \|A^{(n)}(x)v\| \leq \lambda_2(A) \quad \text{if and only if} \quad v \in \nu(x)^\perp,$$
which follows from the facts that the column vectors of
Indeed, the second to last lines of this equation are trivial, and the first line states that
and we also have that
is a cocycle of $T$ [HK02 Proposition 4.1] satisfying $\lambda_2(A) = \lambda_1(D)$ [HK02 Lemma 3.1].
For the sake of self-containedness, we prove the cocycle property here and the equality of
the Lyapunov exponents in Remark 4.5 below. We have
\[ D^{(n)}(x) = \Pi A^{(n)}(x)H(x), \]
with
\[
\Pi = \begin{pmatrix}
0 & 1 & 0 & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & 1 & 0 \\
0 & \cdots & \cdots & 0 & 1
\end{pmatrix},
\]
and we also have that
\[ H(T^n x) \Pi A^{(n)}(x)H(x) = A^{(n)}(x)H(x). \]
Indeed, the second to last lines of this equation are trivial, and the first line states that
\[ -(0, T^n x) A^{(n)}(x) H(x) = (1, 0, \ldots, 0) A^{(n)}(x) H(x), \]
which follows from the facts that the column vectors of $H(x)$ form a basis of $\iota(x)^\perp$ and
\[ A^{(n)}(x) \iota(x)^\perp = \iota(T^n x)^\perp. \]
We obtain that
\[ D^{(1)}(T^n x) D^{(n)}(x) = \Pi A^{(1)}(T^n x) A^{(n)}(x) H(x) = D^{(n+1)}(x), \]
thus $D^{(n)}(x)$ is a cocycle of $T$.

Therefore, it suffices to estimate the first Lyapunov exponent of the cocycle $D^{(n)}(x)$. This is
convenient because it is usually easier to obtain estimates for the first Lyapunov exponent
of a cocycle than for the second one. As observed by Hardcastle and Khanin [Har02 HK02],
the Subadditive Ergodic Theorem yields that
\[ \lambda_2(A) = \lambda_1(D) = \inf_{n \in \mathbb{N}} \frac{1}{n} \int_{\Delta} \log \| D^{(n)}(x) \| \, d\mu(x) \]
for any matrix norm, see [HK02 Lemma 3.3]. We note that the matrices $D^{(n)}$ were first
studied by Fujita, Ito, Keane and Ohtsuki [FIKO96 IKO93]. Observe also that strong
convergence at point $x$ is equivalent to $\lim_{n \to \infty} \| D^{(n)}(x) \| = 0$. Indeed,
\[ \lim_{n \to \infty} \| D^{(n)}(x) \| = 0 \]
means that $\lim_{n \to \infty} |p^{(n)}_{i,j} - q^{(n)}_{i,j} x_j| = 0$ for $i, j \geq 1$, and we then use
the orthogonality of the columns of $A^{(n)}(x) H(x)$ to $\iota(T^n x)$ to deduce that $\lim_{n \to \infty} |p^{(n)}_{0,j} - q^{(n)}_{0,j} x_j| = 0$ for $j \geq 1$.

There exist several methods for providing numerical estimates for the computation of
the second Lyapunov exponent. The approach of [BN96], which is inspired by [JPS87], is
based on the decomposition of matrices as a product of a unitary matrix $Q$ and an upper triangular matrix $R$. For low dimensions $d$, we can also evaluate the integrals in (3.2) symbolically (using polylogarithms) with a computer algebra software such as Mathematica or use estimates for the measure $\mu$ to show that $\lambda_2(A) < 0$ for some continued fraction algorithms, in particular for the Selmer algorithm. Indeed, the densities of invariant measures have simple particular forms; see e.g. (5.1) below. For higher dimensions, these calculations take too much time and we can only make simulations of the behaviour of $D^{(n)}(x)$ for randomly chosen points $x$. According to these simulations, it seems that we have $\lambda_2(A) > 0$ for all known continued fraction algorithms when $d$ gets large, contrary to conjectures of e.g. [Lag93, Har02].

4. On the Paley-Ursell inequality

We recall that the notation $f_n \ll g_n$ means that there exists $C > 0$ such that $f_n \leq C g_n$ for all $n$. Let $D^{(n)}$ be as in (3.1). For certain algorithms, we have

$$\|D^{(n)}(x)\| \ll 1 \text{ uniformly for all } x, n \in \mathbb{N},$$

which can be regarded as a Paley-Ursell type inequality going back to Paley and Ursell [PU30]. Recall that (4.1) means that $|p_{i,j}^{(n)} - q_{i,j}^{(n)} x_j| \ll 1$ for all $i, j \in \{1, \ldots, d\}$ uniformly in $x$ and $n$. In this section we discuss the relations between (4.1) and Paley-Ursell inequality.

We will see in Section 5 that (4.1) holds for Selmer for $d = 2$. It also holds for Brun for $d = 2$ and for Arnoux–Rauzy for arbitrary $d \geq 2$ according to Avila and Delecroix [AD15] and Remark 4.4 below. The original version in [PU30] is proved for Jacobi–Perron with $d = 2$. In the form we state it below, it is contained in Broise and Guivarc’h [BAG01].

Contrary to the results we discussed in the previous section, the results of this section are true for all $x \in \Delta$ (except pathological cases when the algorithm terminates and is not defined). The price we have to pay for getting a result that is valid everywhere is that it is weaker than the metric results we expect to be true. Indeed, while Section 3 is tailored to prove that the second Lyapunov exponent is less than zero almost everywhere, inequality (4.1) implies that $\lambda_2(A) \leq 0$ everywhere. Moreover, (4.1) is true for each time $n$ in an orbit and not only in the limit.

We recall that the notation $\wedge^2$ stands for the second exterior product.

**Proposition 4.1.** Consider a multidimensional continued fraction algorithm satisfying conditions (H1) to (H5). If $\|D^{(n)}(x)\| \ll 1$ holds uniformly in $n \in \mathbb{N}$ and $x$, then

$$\|\wedge^2 A^{(n)}(x)\| \ll \|A^{(n)}(x)\|$$

holds uniformly in $n \in \mathbb{N}$ and $x$.

This result implies that $A^{(n)}(x)$ maps the unit sphere in $\mathbb{R}^n$ to an ellipsoid whose second largest semi-axis $\delta_2(A^{(n)}(x))$ is uniformly bounded in $x \in \Delta$ and $n \in \mathbb{N}$. Moreover, since the elements of $\wedge^2 A^{(n)}(x)$ are the $2 \times 2$ minors of $A^{(n)}(x)$ this inequality shows that the $2 \times 2$ minors of $A^{(n)}(x)$ cannot be much larger than its elements.

To prove this result we need the following preparatory lemma. We write $\delta_i(M)$ for the $i$-th largest singular value of a $d \times d$ matrix $M$ ($1 \leq i \leq d; d \in \mathbb{N}$).
Lemma 4.2. The inequality
\[ \delta_2(A^{(n)}(x)) \ll \delta_1(D^{(n)}(x)) \]
holds uniformly for all \( x \in \Delta \) and all \( n \in \mathbb{N} \).

Proof. Recall that \( D^{(n)}(x) = \Pi A^{(n)}(x) H(x) \). In order to estimate the singular values of \( D^{(n)}(x) \), we map the unit ball \( S^{d-1} \) in \( \mathbb{R}^d \) step by step by the matrices \( H(x), A^{(n)}(x), \) and \( \Pi \), and keep track of the length of the semi-axes of the ellipsoids which are deformed. The ellipsoid \( H(x)S^{d-1} \) is a subset of the hyperplane \( \iota(x) \perp \) whose semi-axes \( a_i^{(1)} \) satisfy \( 1 \ll \|a_i^{(1)}\| \ll 1 \) \((1 \leq i \leq d)\). By the definition of the singular values \( \delta_i(A^{(n)}(x)) \) \((1 \leq i \leq d+1)\), this implies that the ellipse \( A^{(n)}(x)H(x)S^{d-1} \subset \iota(T^n x) \perp \) has semi-axes \( a_i^{(2)} \) satisfying
\[ \|a_i^{(2)}\| \gg \delta_{i+1}(A^{(n)}(x)) \quad (1 \leq i \leq d) \]
(4.3)

It remains to apply the projection \( \Pi \). Since \( \iota(T^n x) = (1, y_1, \ldots, y_d) \) with \( |y_i| \leq 1 \) \((1 \leq i \leq d)\), the angle between the hyperplanes \( \iota(T^n x) \perp \) and \((1, 0, \ldots, 0) \perp \) of \( \mathbb{R}^{d+1} \) is greater than \( c > 0 \) for some constant \( c \) not depending on \( n \). Thus the projection \( \Pi \) shrinks each vector \( v \in \iota(T^n x) \perp \) by a factor which is greater than or equal to \( \sin c \). Thus, because \( A(x)H(x)S^{d-1} \subset \iota(T^n x) \perp \) we get from (4.3) that
\[ \delta_2(A^{(n)}(x)) \ll \delta_1(\Pi A^{(n)}(x) H(x)) = \delta_1(D^{(n)}(x)). \]

We can now finish the proof of Proposition 4.1.

Proof of Proposition 4.1. Suppose that \( \|D^{(n)}(x)\| \ll 1 \) holds. Lemma 4.2 implies that
\[ \| \wedge^2 A^{(n)}(x) \|_2 = \delta_1(\wedge^2 A^{(n)}(x)) = \delta_1(A^{(n)}(x)) \delta_2(A^{(n)}(x)) \ll \delta_1(A^{(n)}(x)) \delta_1(D^{(n)}(x)) = \delta_1(A^{(n)}(x)) \|D^{(n)}(x)\|_2 \ll \|A^{(n)}(x)\|_2, \]
where the implied constants do not depend on \( x \) and \( n \). The estimate in (4.2) follows from this by the equivalence of norms.

We note that the converse of Proposition 4.2 is not true in general. In particular, to get the converse, assumptions on the sequence of matrices \( (A^{(n)}(x))_n \) are needed in order to guarantee that all the quantities \( q^{(i)}_n \) \((0 \leq i \leq d)\) are roughly of the same size for each \( n \) (as is true for instance for the Jacobi–Perron algorithm, see [BAG01, Section 5.2]); see also Proposition 4.3 below. More precisely, one says that the balancedness condition holds for the sequence \( (A^{(n)}(x))_n \) if the norms of the lines of \( A^{(n)}(x) \) are within a multiplicative constant of its norm, with this constant being uniform in \( n \).

Proposition 4.3. Assume that the balancedness condition holds for \( (A^{(n)}(x))_n \). Then
\[ \|D^{(n)}(x)\| \|A^{(n)}(x)\| \ll \| \wedge^2 A^{(n)}(x) \| \quad \text{for all } n. \]
Remark 4.5. For a multidimensional continued fraction algorithm satisfying conditions (H1) to (H5), we recover the fact that λ_1(D) = λ_2(A) from Propositions 4.1 and 4.3, by using that the denominators q_i^{(n)} grow at the same exponential rate, as observed in [Lag93].

5. Selmer Algorithm

5.1. Definition. In its (ordered) homogeneous form, Selmer’s algorithm is defined by subtracting the smallest element from the largest and reordering the elements (see Selmer [Sel61] or Schweiger [Sch00, Chapter 7]), i.e.,

\[ T_S : \Delta \to \Delta, \quad T_S(x_1, \ldots, x_d) \in \mathbb{R} \text{ord}(1 - x_d, x_2, \ldots, x_d). \]

Let

\[ A_S(x) = \begin{cases} S_a & \text{if } x \in \Delta_{S_a} := \{(x_1, \ldots, x_d) \in \Delta : 2x_d > 1\}, \\ S_b & \text{if } x \in \Delta_{S_b} := \{(x_1, \ldots, x_d) \in \Delta : 2x_d < 1 \leq x_{d-1} + x_d\}, \end{cases} \]

with

\[ S_a = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \\ 1 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix}, \quad S_b = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 1 \end{pmatrix}. \]

Since for all \( x \in \Delta \), we have \( T_S^n x \in \Delta_{S_a} \cup \Delta_{S_b} \) for all sufficiently large \( n \) (see [Sch00, Theorem 22]), it suffices to consider the absorbing set \( \Delta_{S_a} \cup \Delta_{S_b} \). Here and in the following, we do not care about the behaviour of \( T_S \) on the boundary of \( \Delta_{S_a} \) and \( \Delta_{S_b} \) because we are interested only in metric results. The invariant measure is

\[ d\mu_S = c \frac{dx_1}{x_1} \frac{dx_2}{x_2} \cdots \frac{dx_d}{x_d} \]
on $\Delta_{S_a} \cup \Delta_{S_b}$, with normalising constant $c$ such that $\mu_\Delta(\Delta_{S_a} \cup \Delta_{S_b}) = 1$; see [Sch00, Theorem 22]. As shown in [Lag93, Section 6], Selmer’s algorithm satisfies the assumptions of Proposition 2.1.

Observe that a multiplicative version of Selmer’s algorithm can also be considered by taking subtractions instead of divisions. It is not an acceleration of the additive version, and does not behave well in terms of convergence [Sch04, Section 2].

5.2. Second Lyapunov exponent, $d = 2$. As mentioned before, Nakashishi [Nak06] gave an intricate proof of the fact that $\lambda_2(A_S) < 0$ for $d = 2$, see also [Sch01b]. We are able to give a very simple proof of this and, on top of this, we bound $\lambda_2(A_S)$ away from 0.

**Theorem 5.1.** For $d = 2$, the second Lyapunov exponent of the Selmer algorithm satisfies

$$\lambda_2(A_S) < -0.050393.$$ 

In particular, the Selmer algorithm is strongly convergent a.e.

**Proof.** We have

$$S_a^2 = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad S_aS_b = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix},$$

and the corresponding matrices $D_S^{(2)}(x_1, x_2)$ are

$$\begin{pmatrix} 1 - x_1 & -x_2 \\ 1 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 - x_2 \\ 1 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 \\ 1 & -x_2 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 \\ 1 - x_1 & 1 - x_2 \end{pmatrix}.$$ 

Since $x_1 + x_2 > 1 > x_1 > x_2 > 0$, we have thus $\|D_S^{(2)}(x)\|_\infty = 1$ for all $x \in \Delta_{S_a} \cup \Delta_{S_b}$. This already implies that $\lambda_1(D_S) \leq 0$ by (3.2).

Moreover, this implies that $\|D_S^{(4)}(x)\|_\infty \leq 1$ for all $x \in \Delta_{S_a} \cup \Delta_{S_b}$. We have

$$(S_aS_b)^2 = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \quad \text{thus } D_S^{(4)}(x_1, x_2) = \begin{pmatrix} 2 - 2x_1 & 1 - 2x_2 \\ 1 - x_1 & 1 - x_2 \end{pmatrix}$$

for $(x_1, x_2) \in \Delta_{S_a} \cap T_S^{-1}\Delta_{S_a} \cap T_S^{-2}\Delta_{S_a} \cap T_S^{-3}\Delta_{S_a}$, i.e., $(x_1, x_2)$ in the triangle with corners $(3/4, 1/2), (3/5, 2/5), (2/3, 1/3)$. We have thus

$$\|D_S^{(4)}(3/4 - \varepsilon, 1/2 - \varepsilon)\|_\infty = 3/4 + 2\varepsilon,$$

hence $\lambda_1(D_S) \leq \frac{1}{4} \int_\Delta \log \|D_S^{(4)}(x)\|_\infty d\mu_\Delta(x) < 0$.

To get better upper bounds for $\lambda_1(D_S)$, note that $A_S^{(n)}(x) = M \in \{S_a, S_b\}^n$ for all $x$ in the triangle

$$\Delta_M = \{x \in \Delta : \iota(x) \in \mathbb{R} \iota(\Delta_{S_a} \cup \Delta_{S_b}) M\}.$$ 

We have thus

$$\lambda_1(D_S) \leq \frac{1}{n} \sum_{M \in \{S_a, S_b\}^n} \mu_\Delta(M) \max_{x \in \Delta_M} \log \|D_S^{(n)}(x)\|_\infty$$
for all $n \geq 1$. The measure of $\Delta_M$ can be calculated using dilogarithms; here we only need to bound it by
\[
\mu_S(\Delta_M) \geq \frac{12}{\pi^2} \min_{x \in \Delta_M} \frac{1}{x_1 x_2} \text{Leb}(\Delta_M) \geq \frac{12}{\pi^2} \min_{x \in \Delta_M} \frac{1}{x_1} \min_{x \in \Delta_M} \frac{1}{x_2} \text{Leb}(\Delta_M);
\]
ote that $c = 12/\pi^2$ in the definition of $\mu_S$ for $d = 2$. Since $\log \|D_S^{(2n)}(x)\|_\infty \leq 0$ for all $x \in \Delta_s \cup \Delta_b$, we obtain, by taking even powers of matrices, that
\[
\lambda_1(D_S) \leq \frac{6}{\pi^2 n} \sum_{M \in \{s, b\}^{2n}} \min_{x \in \Delta_M} \frac{1}{x_1} \min_{x \in \Delta_M} \frac{1}{x_2} \text{Leb}(\Delta_M) \max \log \|D_S^{(2n)}(x)\|_\infty.
\]
As noted in [HK02, Lemma 4.5], the function $x \mapsto \|D_S^{(2n)}(x)\|_\infty$ is convex on $\Delta_M$, hence the maximum is taken on one of the corners of $\Delta_M$. Taking $n = 21$, we obtain that $\lambda_2(A_S) = \lambda_1(D_S) < -0.050393$. This should be compared with the simulations in [Lab15] that suggest that $\lambda_2(A_S) \approx -0.0707$.

In view of Proposition 4.1 we can formulate a result that is true uniformly for all $x \in \Delta$.

**Proposition 5.2.** For the Selmer algorithm with $d = 2$ there exists $C > 0$ such that, for all $x$, $|p^{(n)}_{i,j} - q^{(n)}_{i,j}| \leq C$ for all $i, j \in \{1, \ldots, d\}$. Moreover, the inequality
\[
\|A^{(n)}(x)\| \ll \|A^{(n)}(x)\|
\]holds. Here the implied constant does not depend on $x$ and $n \in \mathbb{N}$.

**Proof.** In the proof of Theorem 5.1 we showed that $\|D^{(2n)}(x)\| \leq 1$. By submultiplicativity this implies that $\|D^{(2n)}(x)\| \leq 1$ and $\|D^{(n)}(x)\| \ll 1$ holds uniformly for all $x \in \Delta$ and all $n \in \mathbb{N}$. The result thus follows from Proposition 4.1.

Avila and Delecroix [AD15] proved that primitive Brun matrices for $d = 2$ and primitive Arnoux–Rauzy matrices with $d \geq 2$ are Pisot, i.e., all eigenvalues except the Perron–Frobenius eigenvalue have absolute value less than 1. We prove the analog result for Selmer with $d = 2$.

**Theorem 5.3.** Let $d = 2$ and $M \in \{s_a, s_b\}^n$ for some $n \geq 1$. The following are equivalent.

1. $M$ is a primitive matrix,
2. $M$ is a Pisot matrix,
3. $M^2 \notin \{s_a s_b, s_b^5\}$.

**Proof.** Let first $M \in \{s_a, s_b\}^n$ be a primitive matrix. Then all eigenvalues of $M$ except the Perron–Frobenius eigenvalue have absolute value $\leq 1$ because $\|D^{(2k)}(x)\|_\infty \leq 1$ for all $x \in \Delta_M$ and $k \geq 1$ by the proof of Theorem 5.1. Since $M$ is a cubic unimodular matrix, this implies that $M$ is a Pisot matrix. On the other hand, it is well known that Pisot matrices are primitive (see e.g. [Fog02, Theorem 1.2.9]), i.e., we have $(1) \iff (2)$.

If $M^2 \in \{s_a s_b, s_b^5\}$, then the first line of $M^{2k}$ equals $(1, 0, 0)$ for all $k \geq 1$, hence $M$ is not primitive (and 1 is an eigenvalue of $M$). Finally, when $M^2 \notin \{s_a s_b, s_b^5\}$, then $M$ is primitive because $M^2$ contains a product $s_a s_b^{2k} s_a$ for some $k \geq 0$, the matrices $s_a^5, s_a s_b$, and $s_b^5$.
\(S^3 a S b a\) and \((S^2 a S b)^2\) are positive, and multiplying by \(S^2 a\) or \((S a S b)^2\) does not decrease any entry of a matrix. This shows that \((1) \iff (3). \square\)

5.3. Second Lyapunov exponent, \(d = 3\). For \(d = 3\) the situation is more intricate than for \(d = 2\). Firstly, \(S a\) has now a pair of complex eigenvalues outside the unit circle, hence we cannot have \(\|D_S^{(n)}(x)\| \leq 1\) for all \(x \in \Delta_S a \cup \Delta_S b\). Secondly, the conjectured value of \(\lambda_2(A_S)\) is approximately \(-0.02283\) and, hence, closer to zero than in the case \(d = 2\). Computer experiments suggest that the smallest value of \(n\) for which the integral in (3.2) can be estimated to be smaller than 0 by the methods used in the proof of Theorem 5.1 is \(n = 52\). We are currently running a program implemented in Objective C parallelizing the crucial computations on GPUs using Metal. Observe also that this indicates that there is no reason for (4.1) and a Paley–Ursell inequality to hold, even when the algorithm \(A\) satisfies \(\lambda_2(A) < 0\).

5.4. Second Lyapunov exponent, \(d \geq 4\). Recall that, for arbitrary \(d\), the cocyle \(D_S^{(n)}(x)\) is given by

\[
D_S^{(1)}(x) = \begin{pmatrix}
0 & 1 & 0 & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & \cdots & 0 & 1 & 0 \\
-x_1 & -x_2 & \cdots & -x_{d-1} & 1 - x_d \\
-x_1 & -x_2 & \cdots & -x_{d-1} & -x_d
\end{pmatrix}
\]

if \(x \in \Delta_S a\), and the last two lines are exchanged for \(x \in \Delta_S b\). (For \(d = 2\), we have \(D_S^{(1)}(x) = \begin{pmatrix}
-x_1 & 1 - x_2 \\
-x_1 & -x_2
\end{pmatrix}\) if \(x \in \Delta_S a\), \(D_S^{(1)}(x) = \begin{pmatrix}
-x_1 & -x_2 \\
-x_1 & 1 - x_2
\end{pmatrix}\) if \(x \in \Delta_S b\).) Evaluating \(\frac{1}{n} \log \|A_S^{(n)}(x)\|\) and \(\frac{1}{n} \log \|D_S^{(n)}(x)\|\) for randomly chosen points \(x\) and \(n = 2^{30}\) gives the estimates listed in Table 1 for \(\lambda_1(A_S)\) and \(\lambda_1(D_S) = \lambda_2(A_S)\) (without guaranteed accuracy).

<table>
<thead>
<tr>
<th>(d)</th>
<th>(\lambda_2(A_S))</th>
<th>(1 - \frac{\lambda_2(A_S)}{\lambda_1(A_S)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-0.07072</td>
<td>1.3871</td>
</tr>
<tr>
<td>3</td>
<td>-0.02283</td>
<td>1.1444</td>
</tr>
<tr>
<td>4</td>
<td>+0.00176</td>
<td>0.9866</td>
</tr>
<tr>
<td>5</td>
<td>+0.01594</td>
<td>0.8577</td>
</tr>
</tbody>
</table>

Table 1. Heuristically estimated values for the second Lyapunov exponent and uniform approximation exponent of the Selmer Algorithm

5.5. Cassaigne algorithm. In 2015, Cassaigne defined an (unordered) continued fraction algorithm that was first studied in [CLL17, AL18] where it was shown to be conjugate to Selmer’s algorithm. The motivation for defining this new algorithm came from word
combinatorics. Define the two matrices
\[
C_a = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \quad C_b = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix},
\]
set \( \Delta' = \{(x_0, x_1, x_2) \in \mathbb{R}_+^3 : x_0 + x_1 + x_2 = 1\} \), and
\[
A_C : \Delta' \rightarrow GL(3, \mathbb{Z}), \quad x \mapsto \begin{cases} C_a & \text{if } x \in \Delta'_{C_a} = \{(x_0, x_1, x_2) \in \Delta' : x_0 > x_2\}, \\ C_b & \text{if } x \in \Delta'_{C_b} = \{(x_0, x_1, x_2) \in \Delta' : x_0 < x_2\}.
\end{cases}
\]
Then the Cassaigne map is
\[
T_C : \Delta' \rightarrow \Delta' \text{ defined by } T_C(x) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ k-1 & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots \\ d-k & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 \end{pmatrix}, \quad 1 \leq k \leq d.
\]
From [CLL17, Section 5], we know that the Cassaigne algorithm is conjugate to the semi-sorted Selmer algorithm (on the absorbing set), which differs from the sorted version of the Selmer algorithm only by the order of the elements. Therefore, all these algorithms have the same Lyapunov spectrum.

6. Brun and modified Jacobi–Perron algorithms

For the Brun algorithm [Bru20, Bru58], the second largest element is subtracted from the largest one, i.e.,
\[
T_B : \Delta \rightarrow \Delta, \quad T_B(x_1, \ldots, x_d) \in \mathbb{R} \text{ord}(1 - x_1, x_2, \ldots, x_d).
\]
To get the associated matrix valued function \( A_B \), we define
\[
B_0 = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ 1 & 1 & \ddots & \ddots & \vdots \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & 1 & 0 \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix}, \quad B_k = \begin{pmatrix} 1 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & \ddots & \vdots & \vdots \\ k-1 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 & 0 \\ 0 & \cdots & \cdots & 0 & 1 \\ 0 & \cdots & \cdots & \cdots & 0 \end{pmatrix}, \quad 1 \leq k \leq d.
\]
Setting \( x_0 = 1, x_{d+1} = 0 \), and
\[
\Delta_{B_k} = \{(x_1, \ldots, x_d) \in \Delta : x_{k+1} < 1 - x_1 < x_k \} \quad (0 \leq k \leq d)
\]
we have
\[
A_B(x) = B_k \quad \text{if } x \in \Delta_{B_k} \quad (0 \leq k \leq d).
\]
In view of [Lag93, Section 6], Brun’s algorithm satisfies the assumptions of Proposition 2.1.
Evaluating \( \frac{1}{n} \log \| A_B^{(n)}(x) \| \) and \( \frac{1}{n} \log \| D_B^{(n)}(x) \| \) for randomly chosen points \( x \) and \( n = 2^{30} \) gives the estimates listed in Table 2 for \( \lambda_1(A_B) \) and \( \lambda_1(D_B) = \lambda_2(A_B) \).

The modified Jacobi–Perron algorithm (or \( d \)-dimensional Gauss algorithm), which goes back to Podsypanin [Pod77], is an accelerated version of the Brun algorithm, defined by the jump transformation \( x \mapsto T_B^n(x) \) with the minimal \( n \geq 1 \) such that \( T_B^{n-1}(x) \notin \Delta_{B_0} \).
Table 2. Heuristically estimated values for the second Lyapunov exponent and the uniform approximation exponent of the Brun Algorithm

<table>
<thead>
<tr>
<th>(d)</th>
<th>(\lambda_2(A_B))</th>
<th>(1 - \frac{\lambda_2(A_B)}{\lambda_1(A_B)})</th>
<th>(d)</th>
<th>(\lambda_2(A_B))</th>
<th>(1 - \frac{\lambda_2(A_B)}{\lambda_1(A_B)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-0.11216</td>
<td>1.3683</td>
<td>7</td>
<td>-0.01210</td>
<td>1.0493</td>
</tr>
<tr>
<td>3</td>
<td>-0.07189</td>
<td>1.2203</td>
<td>8</td>
<td>-0.00647</td>
<td>1.0283</td>
</tr>
<tr>
<td>4</td>
<td>-0.04651</td>
<td>1.1504</td>
<td>9</td>
<td>-0.00218</td>
<td>1.0102</td>
</tr>
<tr>
<td>5</td>
<td>-0.03051</td>
<td>1.1065</td>
<td>10</td>
<td>+0.00115</td>
<td>0.9943</td>
</tr>
<tr>
<td>6</td>
<td>-0.01974</td>
<td>1.0746</td>
<td>11</td>
<td>+0.00381</td>
<td>0.9799</td>
</tr>
</tbody>
</table>

see \cite{Sch00} Section 6.2. Its second Lyapunov exponent is thus negative if and only if \(\lambda_2(A_B) < 0\). In particular, the conjecture of \cite{Har02} that the second Lyapunov exponent is negative for all \(d \geq 2\) seems to be wrong. We mention that for \(d = 2\) the fact that \(\lambda_2(A_B) < 0\) is proved in \cite{IKO93, FIKO96} by heavy use of computer calculation. Later, Meester \cite{Mee99} found a more elegant proof by deriving a Paley–Ursell type inequality for this setting and adapting Schweiger’s argument from \cite{Sch00} Chapter 16. Avila and Delecroix \cite{AD15} gave a simple proof by showing that the \(\infty\)-norm of the restriction of \(A_B^{(n)}(x)\) to \(\iota(x)\perp\) is at most 1; see Remark 4.4. Schratzberger \cite{Sch01a} gave a proof of the strong convergence of Brun algorithm in dimension \(d = 3\). The dependence of the entropy of the Brun algorithm with respect to the dimension is studied in \cite{BLV18}.

7. Jacobi–Perron algorithm

We now consider the Jacobi–Perron algorithm; see \cite{Sch00} Chapter 4 and 16, earlier references are \cite{Ber71, Sch73}. A projective version of this algorithm is given by

\[
T_J : [0, 1]^d \to [0, 1]^d, \quad (x_1, x_2, \ldots, x_d) \mapsto \left(\frac{x_2}{x_1}, \frac{x_3}{x_2}, \ldots, x_d, \frac{1}{x_1} - \left\lfloor \frac{x_2}{x_1} \right\rfloor, \frac{1}{x_1} - \left\lfloor \frac{x_3}{x_2} \right\rfloor, \ldots, \frac{1}{x_1} - \left\lfloor \frac{x_d}{x_{d-1}} \right\rfloor, \frac{1}{x_1}\right).
\]

Its matrix version is therefore

\[
(x_0, x_1, \ldots, x_d) \mapsto \left(\frac{x_1}{x_0}, \frac{x_2}{x_1}, \frac{x_3}{x_2}, \ldots, x_d, \frac{1}{x_1} - \left\lfloor \frac{x_2}{x_1} \right\rfloor, \frac{1}{x_1} - \left\lfloor \frac{x_3}{x_2} \right\rfloor, \ldots, \frac{1}{x_1} - \left\lfloor \frac{x_d}{x_{d-1}} \right\rfloor, \frac{1}{x_1}\right),
\]

and we have

\[
A_J(x_1, \ldots, x_d) = \begin{pmatrix}
\left\lfloor \frac{1}{x_1} \right\rfloor & 1 & \left\lfloor \frac{x_2}{x_1} \right\rfloor & \cdots & \left\lfloor \frac{x_d}{x_{d-1}} \right\rfloor \\
0 & 0 & 1 & 0 & 0 \\
\vdots & \vdots & \ddots & \ddots & 0 \\
0 & \vdots & \ddots & 1 \\
1 & 0 & \cdots & \cdots & 0
\end{pmatrix}.
\]

This is a multiplicative algorithm in the sense that divisions are performed instead of subtractions, hence the coordinates are multiplied by arbitrarily large integers, and there are infinitely many different matrices \(A_J(x)\). It is proved in \cite{Lag93} Section 5 that the Jacobi–Perron algorithm satisfies the assumptions of Proposition 2.1.

It is known that the second Lyapunov exponent of the Jacobi–Perron algorithm is negative for \(d = 2\). A proof of this fact, based on an old result by Paley and Ursell \cite{PU30}, is
given in Schweiger [Sch00, Chapter 16]. Table 3 contains numerical estimates for the Lyapunov exponents of the Jacobi–Perron algorithm for low dimensions. This table indicates that, like for the Brun algorithm, the second Lyapunov exponent of the Jacobi–Perron algorithm is negative for all \( d \leq 9 \) and positive for all \( d \geq 10 \). This gives evidence that [Lag93, Conjecture 1.2] does not hold.

\[
\begin{array}{|c|c|c|c|c|c|}
\hline
\lambda_2(A_j) & 1 - \frac{\lambda_2(A_j)}{\lambda_1(A_j)} & \lambda_2(A_j) & 1 - \frac{\lambda_2(A_j)}{\lambda_1(A_j)} \\
\hline
2 & -0.44841 & 1.3735 & 7 & -0.02819 & 1.0243 \\
3 & -0.22788 & 1.1922 & 8 & -0.01470 & 1.0127 \\
4 & -0.13062 & 1.1114 & 9 & -0.00505 & 1.0044 \\
5 & -0.07880 & 1.0676 & 10 & +0.00217 & 0.9981 \\
6 & -0.04798 & 1.0413 & 11 & +0.00776 & 0.9933 \\
\hline
\end{array}
\]

Table 3. Heuristically estimated values for the second Lyapunov exponent and the uniform approximation exponent of the Jacobi–Perron Algorithm

8. **An intermediate algorithm between Arnoux–Rauzy and Brun**

From [AD15], we know that the second Lyapunov exponent of the Arnoux–Rauzy algorithm is negative for all \( d \geq 2 \), but this algorithm is only defined on a set of Lebesgue measure zero. We propose an algorithm that is in some sense between Arnoux–Rauzy and Brun: We subtract as many of the leading coefficients from the largest one as possible. (In the Arnoux–Rauzy algorithm, we always subtract all but the largest coefficient from the largest one.) The matrix version of this algorithm is (with \( x_{d+1} = x_0 \))

\[
(x_0, x_1, \ldots, x_d) \mapsto \text{ord}\left(x_0 - \sum_{j=1}^{k} x_j, x_1, \ldots, x_d\right) \quad \text{if} \quad \sum_{j=1}^{k} x_j < x_0 < \sum_{j=1}^{k+1} x_j \quad (1 \leq k \leq d).
\]

Denote by \( \Delta_{I_{k,\ell}} \), \( 1 \leq k < d \), \( k \leq \ell \leq d \), the set of \((x_1, \ldots, x_d) \in \Delta \) with \( \sum_{j=1}^{k} x_j < 1 < \sum_{j=1}^{k+1} x_j \) and \( x_\ell > 1 - \sum_{j=1}^{k} x_j > x_{\ell+1} \) (where \( x_{d+1} = 0 \)), and denote by \( \Delta_{I_{d,\ell}} \), \( 0 \leq \ell \leq d \), the set of \((x_1, \ldots, x_d) \in \Delta \) with \( \sum_{j=1}^{d} x_j < 1 \) and \( x_\ell > 1 - \sum_{j=1}^{k} x_j > x_{\ell+1} \) (where \( x_0 = 1 \), \( x_{d+1} = 0 \)). Then we have

\[
A_I(x) = I_{k,\ell} \quad \text{if} \quad x \in \Delta_{I_{k,\ell}}.
\]
The Arnoux–Rauzy algorithm is the special case where \( T^n x \in \Delta_{d,\ell} \), \( 0 \leq \ell \leq d \), for all \( n \geq 0 \). It seems that the second Lyapunov exponent of our intermediate algorithm is negative for all \( d \leq 10 \) and positive for all \( d \geq 11 \). The according heuristic estimates are listed in Table 4.

<table>
<thead>
<tr>
<th>( d )</th>
<th>( \lambda_2(A_I) )</th>
<th>( 1 - \frac{\lambda_2(A_I)}{\lambda_1(A_I)} )</th>
<th>( d )</th>
<th>( \lambda_2(A_I) )</th>
<th>( 1 - \frac{\lambda_2(A_I)}{\lambda_1(A_I)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-0.13648</td>
<td>1.3606</td>
<td>7</td>
<td>-0.02033</td>
<td>1.0729</td>
</tr>
<tr>
<td>3</td>
<td>-0.10803</td>
<td>1.2430</td>
<td>8</td>
<td>-0.01175</td>
<td>1.0468</td>
</tr>
<tr>
<td>4</td>
<td>-0.07540</td>
<td>1.1817</td>
<td>9</td>
<td>-0.00563</td>
<td>1.0246</td>
</tr>
<tr>
<td>5</td>
<td>-0.05035</td>
<td>1.1388</td>
<td>10</td>
<td>-0.00114</td>
<td>1.0054</td>
</tr>
<tr>
<td>6</td>
<td>-0.03263</td>
<td>1.1034</td>
<td>11</td>
<td>+0.00224</td>
<td>0.9886</td>
</tr>
</tbody>
</table>

**Table 4.** Heuristically estimated values for the second Lyapunov exponent and uniform approximation exponent of the intermediate algorithm

Using methods from Messaoudi, Nogueira and Schweiger [MNS09] as well as from Fougeron and Skripchenko [FS19] one can show that the assumptions of Proposition 2.1 hold also for this algorithm. We will come back to this in a forthcoming paper. This will imply that its second Lyapunov exponent is negative implies strong convergence also for this algorithm.

9. **Garrity’s triangle algorithm**

A similar algorithm to the one in Section 8 was proposed by Garrity [Gar01], called the triangle algorithm, with the difference that the smallest coefficient is subtracted as many times as possible from the largest one when all other coefficients have already been subtracted. Similarly as in the case of Selmer’s algorithm (see [Sch04, Section 2]), convergence
properties are altered by taking divisions instead of subtractions. This will be seen on the second Lyapunov exponent below. Observe that this cannot be considered as a real acceleration (as in the regular continued fraction case, or as in the Brun or in the Jacobi–Perron cases), since taking divisions instead of subtractions yields a completely different algorithm (similarly to the Selmer case).

The matrix version of this algorithm is thus

\[
(x_0, x_1, \ldots, x_d) \mapsto \begin{cases}
\text{ord}\left(x_0 - \sum_{j=1}^{k} x_j, x_1, \ldots, x_d\right) & \text{if } \sum_{j=1}^{k} x_j < x_0 < \sum_{j=1}^{k+1} x_j, \ 1 \leq k \leq d - 2, \\
(x_1, \ldots, x_d, x_0 - \sum_{j=1}^{d-1} x_j - \ell x_d) & \text{if } \sum_{j=1}^{d-1} x_j + \ell x_d < x_0 < \sum_{j=1}^{d-1} (\ell + 1)x_d, \ \ell \geq 0.
\end{cases}
\]

We have

\[A_f(x) = G_{k, \ell} \text{ if } x \in \Delta_{G_{k, \ell}},\]

with \(G_{k, \ell} = I_{k, \ell}\) and \(\Delta_{G_{k, \ell}} = \Delta_{I_{k, \ell}}\) for \(1 \leq k \leq d - 2, \ \ell \leq k \leq d,\)

\[
G_{d-1, \ell} = \begin{pmatrix}
1 & 1 & 0 & \cdots & 0 \\
\vdots & 0 & \ddots & \ddots & \vdots \\
\ell & \vdots & \ddots & 1 \\
1 & 0 & \cdots & \cdots & 0
\end{pmatrix}
\text{ for } \ell \geq 0,
\]

\[\Delta_{G_{d-1, \ell}} = \left\{(x_1, \ldots, x_d) \in \Delta : \sum_{j=1}^{d-1} x_j + \ell x_d < 1 < \sum_{j=1}^{d-1} (\ell + 1)x_d \right\}.
\]

Here we have the curious situation that the second Lyapunov exponent seems to be negative if and only if \(7 \leq d \leq 10.\) The according heuristic estimates are listed in Table 6.

<table>
<thead>
<tr>
<th>(d)</th>
<th>(\lambda_2(A_G))</th>
<th>(1 - \frac{\lambda_2(A_G)}{\lambda_1(A_G)})</th>
<th>(d)</th>
<th>(\lambda_2(A_G))</th>
<th>(1 - \frac{\lambda_2(A_G)}{\lambda_1(A_G)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>+0.34434</td>
<td>0.6859</td>
<td>7</td>
<td>-0.00644</td>
<td>1.0225</td>
</tr>
<tr>
<td>3</td>
<td>+0.37673</td>
<td>0.5798</td>
<td>8</td>
<td>-0.00768</td>
<td>1.0304</td>
</tr>
<tr>
<td>4</td>
<td>+0.25232</td>
<td>0.6286</td>
<td>9</td>
<td>-0.00435</td>
<td>1.0189</td>
</tr>
<tr>
<td>5</td>
<td>+0.10677</td>
<td>0.7778</td>
<td>10</td>
<td>-0.00074</td>
<td>1.0035</td>
</tr>
<tr>
<td>6</td>
<td>+0.01859</td>
<td>0.9468</td>
<td>11</td>
<td>+0.00237</td>
<td>0.9880</td>
</tr>
</tbody>
</table>

Table 5. Heuristically estimated values for the second Lyapunov exponent and uniform approximation exponent of Garrity’s simplex algorithm.

Again using methods from [MNS09] and [FS19] one can show that the assumptions of Proposition 2.1 hold also for this algorithm in any dimension (although this is a bit more involved in this case because the algorithm is multiplicative); the case \(d = 2\) is handled in [FS19]; also this will be addressed in a forthcoming paper. This will imply that its second Lyapunov exponent is negative implies strong convergence also for this algorithm.
10. Heuristical comparison between the algorithms

We conclude with a table that allows to compare the (heuristically estimated) uniform approximation coefficients of the algorithms considered in this paper. We also indicate below Dirichlet’s bound, namely $1 + 1/d$.

<table>
<thead>
<tr>
<th>$d$</th>
<th>Selmer</th>
<th>Brun</th>
<th>Jacobi–Perron</th>
<th>Intermediate</th>
<th>Garry</th>
<th>$1+1/d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1.3871</td>
<td>1.3683</td>
<td>1.3735</td>
<td>1.3606</td>
<td>0.6859</td>
<td>1.5</td>
</tr>
<tr>
<td>3</td>
<td>1.1444</td>
<td>1.2203</td>
<td>1.1922</td>
<td>1.2430</td>
<td>0.5798</td>
<td>1.3333</td>
</tr>
<tr>
<td>4</td>
<td>0.9866</td>
<td>1.1504</td>
<td>1.1114</td>
<td>1.1817</td>
<td>0.6268</td>
<td>1.25</td>
</tr>
<tr>
<td>5</td>
<td>0.8577</td>
<td>1.1065</td>
<td>1.0676</td>
<td>1.1388</td>
<td>0.7778</td>
<td>1.2</td>
</tr>
<tr>
<td>6</td>
<td>0.7442</td>
<td>1.0746</td>
<td>1.0413</td>
<td>1.1034</td>
<td>0.9468</td>
<td>1.1667</td>
</tr>
<tr>
<td>7</td>
<td>0.6437</td>
<td>1.0493</td>
<td>1.0243</td>
<td>1.0729</td>
<td>1.0225</td>
<td>1.1429</td>
</tr>
<tr>
<td>8</td>
<td>0.5561</td>
<td>1.0283</td>
<td>1.0127</td>
<td>1.0468</td>
<td>1.0304</td>
<td>1.125</td>
</tr>
<tr>
<td>9</td>
<td>0.4810</td>
<td>1.0102</td>
<td>1.0044</td>
<td>1.0246</td>
<td>1.0189</td>
<td>1.1111</td>
</tr>
<tr>
<td>10</td>
<td>0.4173</td>
<td>0.9943</td>
<td>0.9981</td>
<td>1.0054</td>
<td>1.0035</td>
<td>1.1</td>
</tr>
<tr>
<td>11</td>
<td>0.3636</td>
<td>0.9799</td>
<td>0.9933</td>
<td>0.9886</td>
<td>0.9880</td>
<td>1.0909</td>
</tr>
</tbody>
</table>

Table 6. Uniform approximation exponents $1 - \frac{\lambda_2(A)}{\lambda_1(A)}$

References


Université de Paris, IRIF, CNRS, F–75013 Paris, FRANCE
Email address: berthe@irif.fr, steiner@irif.fr

Chair of Mathematics and Statistics, University of Leoben, A–8700 Leoben, AUSTRIA
Email address: joerg.thuswaldner@unileoben.ac.at