FINDING SIMULTANEOUS DIOPHANTINE APPROXIMATIONS WITH PRESCRIBED QUALITY

WIEB BOSMA AND IONICA SMEETS

ABSTRACT. We give an algorithm that finds a sequence of approximations with Dirichlet coefficients bounded by a constant only depending on the dimension. The algorithm uses the LLL-algorithm for lattice basis reduction. We present a version of the algorithm that runs in polynomial time of the input.

1. Introduction

The regular continued fraction algorithm is a classical algorithm to approximate reals by rational numbers. Dirichlet [15] proved that for every $a \in \mathbb{R}$ there are infinitely many integers q such that

$$\|q\, a\| < q^{-1}, \tag{1}$$

where $\|x\|$ denotes the distance between x and the nearest integer. The exponent -1 of q is minimal; if it is replaced by any number $e < -1$, then there exist real numbers a such that only finitely many integers q satisfy $\|q\, a\| < q^e$.

Hurwitz [8] proved that the continued fraction algorithm finds, for every $a \in \mathbb{R} \setminus \mathbb{Q}$, an infinite sequence of increasing integers q_n with

$$\|q_n\, a\| < \frac{1}{\sqrt{5}}\, q_n^{-1}.$$

If the constant $\frac{1}{\sqrt{5}}$ is replaced by any smaller one, then this statement is false. Legendre [14] showed that the continued fraction algorithm finds all good approximations, in the sense that if for some positive integer q

$$\|q\, a\| < \frac{1}{2}\, q^{-1},$$

then q is one of the q_n found by the algorithm.

As to the generalization of approximations in higher dimensions Dirichlet proved the following theorem; see Chapter II of [18].

Theorem 2. *Let an $n \times m$ matrix A with entries $a_{ij} \in \mathbb{R} \setminus \mathbb{Q}$ be given and suppose that $1, a_{i1}, \dots, a_{im}$ are linearly independent over $\mathbb{Q}$ for some i with $1 \le i \le n$. There exist infinitely many coprime m-tuples of integers $(q_1, \dots, q_m)$ such that with $q = \max_j |q_j| \ge 1$, we have*

$$\max_i \|q_1\, a_{i1} + \cdots + q_m\, a_{im}\| < q^{-\frac{m}{n}}. \tag{3}$$

If the exponent $-\frac{m}{n}$ is replaced by any smaller number, there exists a matrix A for which the inequality holds for finitely many coprime tuples $(q_1, q_2, \ldots, q_m)$ only.

Definition 4. Let an $n \times m$ matrix A with entries $a_{ij} \in \mathbb{R} \setminus \mathbb{Q}$ be given. The *Dirichlet coefficient* of an m-tuple $(q_1, \ldots, q_m)$ is $q^{\frac{m}{n}} \max_i \|q_1\, a_{i1} + \cdots + q_m\, a_{im}\|$.

The proof of the theorem does not give an efficient way of finding a series of approximations with a Dirichlet coefficient less than 1. For the case $m = 1$ the first multi-dimensional continued fraction algorithm was given by Jacobi [9]. Many more followed, see for instance Perron [17], Brun [5], Lagarias [13] and Just [10]. Brentjes [4] gives a detailed history and description of such algorithms. Schweiger's book [19] gives a broad overview. For $n = 1$ there is, amongst others, the algorithm by Ferguson and Forcade [7]. However, there is no efficient algorithm guaranteed to find a series of approximations with Dirichlet coefficient smaller than 1.

In 1982 the LLL-algorithm for lattice basis reduction was published in [16]. The authors noted that their algorithm could be used for finding Diophantine approximations of given rationals with Dirichlet coefficient only depending on the dimension; see Corollary 14. Just [10] developed an algorithm based on lattice reduction that detects $\mathbb{Z}$-linear dependence in the a_i, in the case $m = 1$. If no such dependence is found her algorithm returns integers q with

$$\max_i \|qa_i\| \leq c \left(\sum_{i=1}^{n} a_i^2 \right)^{1/2} q^{-1/(2n(n-1))},$$

where c is a constant depending on n. The exponent $-1/(2n(n-1))$ is larger than the Dirichlet exponent $-1/n$. Lagarias [12] used the LLL-algorithm in a series of lattices to find good approximations for the case $m = 1$. Let $a_1, \ldots, a_n \in \mathbb{Q}$ and let N be a positive integer; suppose there exists $Q \in \mathbb{N}$ with $1 \leq Q \leq N$ such that $\max_j ||Q\, a_j|| < \varepsilon$. Then Lagarias' algorithm on input $a_1, \ldots, a_n$ and N finds in polynomial time a q with $1 \leq q \leq 2^{\frac{n}{2}} N$ such that $\max_j ||q\, a_j|| \leq \sqrt{5n} 2^{\frac{n-1}{2}} \varepsilon$. One difference with our work is that Lagarias focuses on the quality $||q\, a_j||$, while we focus on the Dirichlet coefficient $q^{\frac{1}{n}} ||q\, a_j||$. We also consider the case $m > 1$.

The main result of the present paper is an algorithm that by iterating the LLL-algorithm gives a series of approximations of given rationals with optimal Dirichlet exponent. Where the LLL-algorithm gives one approximation, our dynamic algorithm gives a series of successive approximations. To be more precise: for a given $n \times m$-matrix A with entries $a_{ij} \in \mathbb{Q}$ and a given upper bound $q_{\max}$ the algorithm returns a sequence of m-tuples $(q_1, \ldots, q_m)$ such that for every Q with $2^{\frac{(m+n+3)(m+n)}{4m}} \leq Q \leq q_{\max}$ one of these m-tuples satisfies

$$\max_j |q_j| \leq Q \quad \text{and} \quad \max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| \leq 2^{\frac{(m+n+3)(m+n)}{4n}} Q^{-\frac{m}{n}}.$$

The exponent $-\frac{m}{n}$ of Q can not be improved and therefore we say that these approximations have optimal Dirichlet exponent.

Our algorithm is a multi-dimensional continued fraction algorithm in the sense that we work in a lattice basis and that we only interchange basis vectors and add integer multiples of basis vectors to another. Our algorithm differs from other multi-dimensional continued fraction algorithms in that the lattice is not fixed across iterations. In Lemma 26 we show that if there exists an extremely good approximation, our algorithm finds a very good one. We derive in Theorem 36 how the output of our algorithm gives a lower bound on the quality of possible approximations with coefficients up to a certain limit. In Section 4 we show that a slightly modified version of our algorithm runs in polynomial time. In Section 5 we present some numerical data.

An earlier version of this paper appeared as Chapter V of the thesis of the second author [20]. Some style and numbering options were adopted from this.

2. Lattice reduction and the LLL-algorithm

In this section we give the definitions and results that we need for our algorithm.

Let r be a positive integer. A subset L of the r-dimensional real Euclidean vector space $\mathbb{R}^r$ is called a *lattice* if there exists a basis $b_1, \ldots, b_r$ of $\mathbb{R}^r$ such that

$$L = \sum_{i=1}^{r} \mathbb{Z} b_i = \left\{ \sum_{i=1}^{r} z_i b_i ; z_i \in \mathbb{Z}\,(1 \le i \le r) \right\}.$$

We say that $b_1, \ldots, b_r$ is a *basis* for L. The *determinant* of the lattice L is defined by $|\det(b_1, \ldots, b_r)|$ and we denote it as $\det(L)$.

For any linearly independent $b_1, \ldots, b_r \in \mathbb{R}^r$ the Gram-Schmidt process yields an orthogonal basis $b_1^*, \ldots, b_r^*$ for $\mathbb{R}^r$, by defining inductively

$$\begin{aligned} b_i^* &= b_i - \sum_{j=1}^{i-1} \mu_{ij} b_j^* \quad \text{for } 1 \le i \le r \quad \text{and} \\ \mu_{ij} &= \frac{(b_i, b_j^*)}{(b_j^*, b_j^*)}, \end{aligned} \tag{5}$$

where $(\,,)$ denotes the ordinary inner product on $\mathbb{R}^r$.

We call a basis $b_1, \ldots, b_r$ for a lattice L *reduced* if

$$|\mu_{ij}| \le \frac{1}{2} \quad \text{for } 1 \le j < i \le r$$

and

$$|b_i^* + \mu_{ii-1} b_{i-1}^*|^2 \le \frac{3}{4} |b_{i-1}^*|^2 \quad \text{for } 1 \le i \le r,$$

where $|x|$ denotes the Euclidean length of x.

The following two propositions were proven in [16].

Proposition 6. *Let $b_1, \ldots, b_r$ be a reduced basis for a lattice L in $\mathbb{R}^r$. Then we have*

$$\begin{aligned} &(i) && |b_1| \le 2^{(r-1)/4} \left(\det(L)\right)^{1/r}, \\ &(ii) && |b_1|^2 \le 2^{r-1}\,|x|^2, \quad \textit{for every } x \in L, x \ne 0, \\ &(iii) && \prod_{i=1}^{r} |b_i| \le 2^{r(r-1)/4} \det(L). \end{aligned}$$

Proposition 7. *Let $L \subset \mathbb{Z}^r$ be a lattice with a basis $b_1, b_2, \ldots, b_r$, and let $F \in \mathbb{R}$, $F \ge 2$, be such that $|b_i|^2 \le F$ for $1 \le i \le r$. Then the number of arithmetic operations needed by the LLL-algorithm is $O(r^4 \log F)$ and the integers on which these operations are performed each have binary length $O(r \log F)$.*

In the following Lemma the approach suggested in the original LLL-paper for finding (simultaneous) Diophantine approximations is generalized to the case $m > 1$.

Lemma 8. *Let an $n \times m$-matrix A with entries $a_{ij} \in \mathbb{R}$ and an $\varepsilon \in (0,1)$ be given. Let L be the lattice formed by the columns of the $(m+n) \times (m+n)$-matrix*

$$B = \begin{bmatrix} 1 & 0 & \ldots & 0 & a_{11} & \ldots & a_{1m} \\ 0 & 1 & \ddots & 0 & a_{21} & \ldots & a_{2m} \\ \vdots & & & \vdots & \vdots & & \vdots \\ 0 & \ldots & 0 & 1 & a_{n1} & \ldots & a_{nm} \\ 0 & \ldots & 0 & 0 & c & & 0 \\ \vdots & & \vdots & \vdots & & \ddots & \\ 0 & \ldots & 0 & 0 & 0 & & c \end{bmatrix}, \tag{9}$$

with $c = \left(2^{-\frac{m+n-1}{4}} \varepsilon\right)^{\frac{m+n}{m}}$.

The LLL-algorithm applied to L will yield an m-tuple $(q_1, \ldots, q_m)$ of integers with

$$\max_j |q_j| \quad \le \quad 2^{\frac{(m+n-1)(m+n)}{4m}} \varepsilon^{-\frac{n}{m}} \ \textit{and} \tag{10}$$

$$\max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| \quad \le \quad \varepsilon. \tag{11}$$

Proof. The LLL-algorithm finds a reduced basis $b_1, \ldots, b_{m+n}$ for the lattice L. For each vector b in this basis there exist $p_i \in \mathbb{Z}$, for $1 \le i \le n$, and $q_j \in \mathbb{Z}$, for $1 \le j \le m$, such that

$$b = \begin{bmatrix} q_1 a_{11} + \cdots + q_m a_{1m} - p_1 \\ \vdots \\ q_1 a_{n1} + \cdots + q_m a_{nm} - p_n \\ c q_1 \\ \vdots \\ c q_m \end{bmatrix}.$$

Proposition 6(i) gives an upper bound for the length of the first basis vector,

$$|b_1| \le 2^{\frac{m+n-1}{4}} c^{\frac{m}{m+n}}.$$

From this vector b_1 we find integers $q_1, \dots, q_m$, such that

$$\max_j |q_j| \leq 2^{\frac{m+n-1}{4}} c^{\frac{-n}{m+n}} \text{ and} \tag{12}$$

$$\max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| \leq 2^{\frac{m+n-1}{4}} c^{\frac{m}{m+n}}. \tag{13}$$

Substituting $c = \left(2^{-\frac{m+n-1}{4}} \varepsilon\right)^{\frac{m+n}{m}}$ gives the results. □

From equations (12) and (13) we obtain the following corollary.

Corollary 14. *For any $n \times m$-matrix A with entries $a_{ij} \in \mathbb{R}$ the LLL-algorithm can be used to obtain an m-tuple $(q_1, \dots, q_m)$ that satisfies, with $q = \max_j |q_j|$,*

$$\max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| \leq 2^{\frac{(m+n-1)(m+n)}{4n}} q^{-\frac{m}{n}}. \tag{15}$$

3. The Iterated LLL-algorithm

We iterate the LLL-algorithm over a series of lattices to find a sequence of approximations. We start with a lattice determined by a basis of the form (9). After the LLL-algorithm finds a reduced basis for this lattice, we decrease the constant c by dividing the last m rows of the matrix by a constant d greater than 1. By doing so, ε is divided by $d^{\frac{m}{m+n}}$. We repeat this process until the upper bound (10) for $\max |q_j|$ guaranteed by the LLL-algorithm exceeds a given upper bound $q_{\max}$.

To ease notation we put $d = 2$ and $\varepsilon = \frac{1}{2}$.

Iterated LLL-algorithm (ILLL)

Input

An $n \times m$-matrix A with entries a_{ij} in $\mathbb{R}$.
An upper bound $q_{\max} > 1$.

Output

For each integer k with $1 \leq k \leq k'$, see (18), we obtain a vector $q(k) \in \mathbb{Z}^m$ with

$$\max_j |q_j(k)| \leq 2^{\frac{(m+n-1)(m+n)}{4m}} 2^{\frac{kn}{m}}, \tag{16}$$

$$\max_i \|q_1(k)\, a_{i1} + \cdots + q_m(k)\, a_{im}\| \leq \frac{1}{2^k}. \tag{17}$$

Description of the algorithm

(1) Construct the basis matrix B as given in (9) from A.
(2) Apply the LLL-algorithm to B.
(3) Deduce $q_1, \dots, q_m$ from the first vector in the reduced basis returned by the LLL-algorithm.
(4) Divide the last m rows of B by $2^{\frac{m+n}{m}}$
(5) Stop if the upper bound for q guaranteed by the algorithm (16) exceeds $q_{\max}$; else go to Step 2.

Define

$$k' := \left\lceil -\frac{(m+n-1)(m+n)}{4n} + \frac{m \log_2 q_{\max}}{n} \right\rceil. \tag{18}$$

Remark 19. The number $2^{\frac{m+n}{m}}$ in Step 4 may be replaced by $d^{\frac{m+n}{m}}$ for any real number $d > 1$. When we additionally set $\varepsilon = \frac{1}{d}$ this yields that

$$\max_j |q_j(k)| \quad \leq \quad 2^{\frac{(m+n-1)(m+n)}{4m}} d^{\frac{kn}{m}} \qquad \text{and} \tag{20}$$

$$\max_i \|q_1(k)a_{i1} + \cdots + q_m(k)a_{im}\| \quad < \quad d^{-k}. \tag{21}$$

In this paper, with the exception of the numerical examples in Section 5, we always take $d = 2$ and $\varepsilon = \frac{1}{2}$.

Lemma 22. *Let an $n \times m$-matrix A with entries a_{ij} in $\mathbb{R}$ and an upper bound $q_{\max} > 1$ be given. With this input, the number of times the ILLL-algorithm applies the LLL-algorithm equals k' from* (18).

Proof. One derives the number of iterations by solving k from the stopping criterion (16)

$$q_{\max} \leq 2^{\frac{(m+n-1)(m+n)}{4m}} 2^{\frac{kn}{m}},$$

that is:

$$\frac{m}{n} \log_2 q_{\max} \leq \frac{(m+n-1)(m+n)}{4n} + k.$$

We stop iterating as soon as the integer k reaches the ceiling k' as in (18). $\square$

We define

$$c(k) = c(k-1)/2^{\frac{m+n}{m}} \text{ for } k > 1, \qquad \text{where } c(1) = c \text{ as given in Lemma 8.}$$

In iteration k we are working in the lattice defined by the basis in (9) with c replaced by $c(k)$.

Lemma 23. *The output $q(k) = (q_1(k), q_2(k), \ldots, q_m(k))$ of the ILLL-algorithm satisfies* (16) *and* (17)*, for $\leq k \leq k'$.*

Proof. In the k-th iteration we use $c(k) = \left(2^{-\frac{m+n+3}{4}-k+1}\right)^{\frac{m+n}{m}}$. Substituting $c(k)$ for c in equations (12) and (13) yields (16) and (17), respectively. $\square$

The following theorem gives the main result of the present paper, as mentioned in the introduction. The algorithm returns a sequence of approximations with all coefficients smaller than Q, optimal Dirichlet exponent and Dirichlet coefficient only depending on the dimensions m and n.

Theorem 24. *Let an $n \times m$-matrix A with entries a_{ij} in $\mathbb{R}$, and $q_{\max} > 1$ be given. The ILLL-algorithm finds a sequence of m-tuples $(q_1, \ldots, q_m)$ of integers such that for every Q with $2^{\frac{(m+n+3)(m+n)}{4m}} \leq Q \leq q_{\max}$ one of these m-tuples satisfies*

$$\max_j |q_j| \leq Q \qquad \text{and} \qquad \max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| \leq 2^{\frac{(m+n+3)(m+n)}{4n}} Q^{-\frac{m}{n}}.$$

Proof. Take $k \in \mathbb{N}$ such that

$$2^{\frac{(m+n+3)(m+n)}{4m}} \cdot 2^{\frac{(k-1)n}{m}} \leq Q < 2^{\frac{(m+n+3)(m+n)}{4m}} \cdot 2^{\frac{kn}{m}}. \tag{25}$$

From Lemma 23 we know that $q(k) = (q_1(k), q_2(k), \ldots, q_m(k))$ satisfies the inequality

$$\max_j |q_j(k)| \leq 2^{\frac{(m+n+3)(m+n)}{4m}} 2^{\frac{(k-1)n}{m}} \leq Q.$$

From the right hand side of inequality (25) if follows that $\frac{1}{2^k} < 2^{\frac{(m+n+3)(m+n)}{4n}} Q^{-\frac{m}{n}}$. From Lemma 23 and this inequality we derive that

$$\max_i \|q_1(k)\, a_{i1} + \cdots + q_m(k)\, a_{im}\| \leq \frac{1}{2^k} < 2^{\frac{(m+n+3)(m+n)}{4n}} Q^{-\frac{m}{n}}.$$

□

Proposition 6(ii) guarantees that if there exists an extremely short vector in the lattice, then the LLL-algorithm finds a rather short lattice vector. We extend this result to the realm of successive approximations. In the next lemma we show that for every very good approximation (satisfying (28)), the ILLL-algorithm finds a rather good one (satisfying (31)) not too far away from it (as specified by (30)).

Lemma 26. *Let an $n \times m$-matrix A with entries a_{ij} in $\mathbb{R}$, a real number $0 < \delta < 1$ and an integer $s > 1$ be given. If there exists an m-tuple $(s_1, \ldots, s_m)$ of integers with*

$$s = \max_j |s_j| > 2^{\frac{(m+n-1)n}{4m}} \left(\frac{n\delta^2}{m}\right)^{\frac{n}{2(m+n)}} \tag{27}$$

and

$$\max_i \|s_1 a_{i1} + \cdots + s_m a_{im}\| \leq \delta s^{\frac{-m}{n}}, \tag{28}$$

then applying the ILLL-algorithm with

$$q_{\max} \geq 2^{\frac{m^2+m(n-1)+4n}{4m}} \left(\frac{m}{n\delta^2}\right)^{\frac{n}{2(m+n)}} s \tag{29}$$

yields an m-tuple $(q_1, \ldots, q_m)$ of integers with

$$\max_j |q_j| \leq 2^{\frac{m^2+m(n-1)+4n}{4m}} \left(\frac{m}{n\delta^2}\right)^{\frac{n}{2(m+n)}} s \tag{30}$$

and

$$\max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| \leq 2^{\frac{m+n}{2}} \sqrt{n}\delta s^{\frac{-m}{n}}. \tag{31}$$

Proof. Let $1 \leq k \leq k'$ be an integer. Proposition 6(ii) gives that for each m-tuple $q(k)$ found by the algorithm

$$\sum_{i=1}^{n} \|q_1(k)a_{i1} + \cdots + q_m(k)a_{im}\|^2 + c(k)^2 \sum_{j=1}^{m} q_j(k)^2$$

$$\leq 2^{m+n-1} \left(\sum_{i=1}^{n} \|s_1 a_{11} + \cdots + s_m a_{im}\|^2 + c(k)^2 \sum_{j=1}^{m} s_j^2 \right).$$

From this and (27) and (28) it follows that

$$\max_i \|q_1(k)a_{i1} + \cdots + q_m(k)a_{im}\|^2 \leq 2^{m+n-1}\left(n\delta^2 s^{\frac{-2m}{n}} + c(k)^2 ms^2\right). \tag{32}$$

Take the smallest positive integer K such that

$$c(K) \leq \sqrt{\frac{n}{m}}\delta s^{-\frac{m+n}{n}}. \tag{33}$$

We find for the K-th iteration from (32) and (33)

$$\max_i \|q_1(K)a_{i1} + \cdots + q_m(K)a_{im}\| \leq 2^{\frac{m+n}{2}}\sqrt{n}\delta s^{\frac{-m}{n}},$$

which gives (31).

We show that under assumption (29) the ILLL-algorithm goes through at least K iterations. We may assume $K > 1$, since the ILLL-algorithm does always at least 1 iteration. From Lemma 22 we find that if $q_{\max}$ satisfies

$$q_{\max} > 2^{\frac{Kn}{m}} 2^{\frac{(m+n-1)(m+n)}{4m}},$$

then the ILLL-algorithm does at least K iterations. Our choice of K implies

$$c(K-1) = \frac{c(1)}{2^{\frac{(m+n)(K-2)}{m}}} = \frac{2^{-\frac{(m+n+3)(m+n)}{4m}}}{2^{\frac{(m+n)(K-2)}{m}}} > \sqrt{\frac{n}{m}}\delta s^{-\frac{m+n}{n}},$$

and we obtain

$$2^{\frac{Kn}{m}} < 2^{-\frac{(m+n-5)n}{4m}}\left(\frac{m}{n\delta^2}\right)^{\frac{n}{2(m+n)}} s.$$

From this we find that

$$q_{\max} > 2^{\frac{m^2+m(n-1)+4n}{4m}}\left(\frac{m}{n\delta^2}\right)^{\frac{n}{2(m+n)}} s$$

is a satisfying condition to guarantee that the algorithm does at least K iterations.

Furthermore, either $2^{\frac{-(m+n)}{m}}\sqrt{\frac{n}{m}}\delta s^{-\frac{m+n}{n}} < c(K)$ or $K = 1$. In the former case we find from (12) that

$$\max_j |q_j(K)| \leq 2^{\frac{m+n-1}{4}} c(K)^{\frac{-n}{m+n}} < 2^{\frac{m+n-1}{4}} 2^{\frac{n}{m}}\left(\frac{m}{n\delta^2}\right)^{\frac{n}{2(m+n)}} s.$$

In the latter case we obtain from (12)

$$\max_j |q_j(1)| \leq 2^{\frac{m+n-1}{4}} c(1)^{\frac{-n}{m+n}} = 2^{\frac{m+n-1}{4}} 2^{\frac{(m+n+3)n}{4m}}$$

and, by (27),

$$2^{\frac{m+n-1}{4}} 2^{\frac{(m+n+3)n}{4m}} = 2^{\frac{m+n-1}{4}} 2^{\frac{n}{m}} 2^{\frac{(m+n-1)n}{4m}} < 2^{\frac{m+n-1}{4}} 2^{\frac{n}{m}}\left(\frac{m}{n\delta^2}\right)^{\frac{n}{2(m+n)}} s.$$

We conclude that for all $K \geq 1$

$$\max_j |q_j(K)| \leq 2^{\frac{m^2+m(n-1)+4n}{4m}}\left(\frac{m}{n\delta^2}\right)^{\frac{n}{2(m+n)}} s.$$

□

From equations (30) and (31) we obtain the following corollary.

Corollary 34. *With the assumptions of Lemma 26, the ILLL-algorithm can be used to obtain an m-tuple $(q_1, \dots, q_m)$ of integers that satisfies*

$$q^{\frac{m}{n}} \max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| \le 2^{\frac{m^2+m(3n-1)+4n+2n^2}{4n}} m^{\frac{m}{2(m+n)}} (n\delta^2)^{\frac{n}{2(m+n)}}, \tag{35}$$

where again $q = \max_j |q_j|$.

Theorem 36. *Let an $n \times m$-matrix A with entries a_{ij} in $\mathbb{R}$ and $q_{\max} > 1$ be given. Assume that γ is such that for every m-tuple $(q_1, \dots, q_m)$ returned by the ILLL-algorithm*

$$q^{\frac{m}{n}} \max_i \|q_1 a_{i1} + \dots q_m a_{im}\| > \gamma, \text{ where } q = \max_j |q_j|. \tag{37}$$

Then every m-tuple $(s_1, \dots, s_m)$ of integers with $s = \max_j |s_j|$ and

$$2^{\frac{(m+n-1)n}{4m}} \left(\frac{n\delta^2}{m}\right)^{\frac{n}{2(m+n)}} < s < 2^{-\frac{m^2+m(n-1)+4n}{4m}} \left(\frac{n\delta^2}{m}\right)^{\frac{n}{2(m+n)}} q_{\max}$$

satisfies

$$s^{\frac{m}{n}} \max_i \|s_1 a_{i1} + \cdots + s_m a_{im}\| > \delta,$$

with

$$\delta = 2^{\frac{-(m+n)(m^2+m(3n-1)+4n+2n^2)}{4n^2}} m^{\frac{-m}{2n}} n^{\frac{-1}{2}} \gamma^{\frac{m+n}{n}}. \tag{38}$$

Proof. Assume that every vector returned by our algorithm satisfies (37) and that there exists an m-tuple $(s_1, \dots, s_m)$ with $s = \max_j |s_j|$ such that

$$2^{\frac{(m+n-1)n}{4m}} \left(\frac{n\delta^2}{m}\right)^{\frac{n}{2(m+n)}} < s < 2^{-\frac{m^2+m(n-1)+4n}{4m}} \left(\frac{n\delta^2}{m}\right)^{\frac{n}{2(m+n)}} q_{\max}$$
$$\text{and} \quad s^{\frac{m}{n}} \max_i \|s_1 a_{i1} + \cdots + s_m a_{im}\| \le \delta.$$

From the upper bound on s it follows that q_{max} satisfies (29). We apply Lemma 26 and find that the algorithm finds an m-tuple $(q_1, \dots, q_m)$ that satisfies (35). Substituting δ as given in (38) gives

$$q^{\frac{m}{n}} \max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| \le \gamma,$$

which is a contradiction with our assumption. □

4. A polynomial time version of the ILLL-algorithm

We have used real numbers in our theoretical results, but in a practical implementation of the algorithm we only use rational numbers. Without loss of generality we may assume that these numbers are in the interval $[0, 1]$. In this section we describe the changes to the algorithm and we show that this modified version of the algorithm runs in polynomial time.

As input for the rational algorithm we take

- the dimensions m and n,
- a rational number $\varepsilon \in (0, 1)$,
- an integer M that is large compared to $\frac{(m+n)^2}{m} - \frac{m+n}{m} \log \varepsilon$,

- an $n \times m$-matrix A with entries $0 < a_{ij} \leq 1$, where each $a_{ij} = \frac{p_{ij}}{2^M}$ for some integer p_{ij},
- an integer $q_{\max} < 2^M$.

Remark 39. In this rational algorithm all irrational numbers are approximated by rational numbers with denominator 2^M. Thus M denotes the precision that is used.

When we construct the matrix B in Step 1 of the ILLL-algorithm we approximate c as given in (9) by a rational number

$$\hat{c} = \frac{\lceil 2^M c \rceil}{2^M} = \frac{\left\lceil 2^M \left(2^{-\frac{m+n-1}{4}} \varepsilon\right)^{\frac{m+n}{m}} \right\rceil}{2^M}. \tag{40}$$

Hence $c < \hat{c} \leq c + \frac{1}{2^M}$.

In iteration k we use a rational $\hat{c}(k)$ that for $k \geq 2$ is given by

$$\hat{c}(k) = \frac{\left\lceil 2^M \hat{c}(k-1) 2^{-\frac{m+n}{m}} \right\rceil}{2^M} \text{ and } \hat{c}(1) = \hat{c} \text{ as in (40),}$$

and we change Step 4 of the ILLL-algorithm to 'multiply the last m rows of B by $\hat{c}(k-1)/\hat{c}(k)$'. The other steps of the rational iterated algorithm are as described in Section 3.

4.1. **The running time of the rational algorithm.**

Theorem 41. *Let the input be given as described above. Then the number of arithmetic operations needed by the ILLL-algorithm and the binary length of the integers on which these operations are performed are both bounded by a polynomial in m, n and M.*

Proof. The number of times we apply the LLL-algorithm is not changed by rationalizing c, so we find the number of iterations k' from Lemma 22

$$k' = \left\lceil -\frac{(m+n-1)(m+n)}{4n} + \frac{m \log_2 q_{\max}}{n} \right\rceil < \left\lceil \frac{mM}{n} \right\rceil.$$

It is obvious that Steps 1, 3, 4 and 5 of the algorithm are polynomial in the size of the input and we focus on the LLL-step. We determine an upper bound for the length of a basis vector used at the beginning of an iteration in the ILLL-algorithm.

In the first application of the LLL-algorithm the length of the initial basis vectors as given in (9) is bounded by

$$|b_i|^2 \leq \max_j \left\{1, a_{1j}^2 + \cdots + a_{nj}^2 + m\hat{c}^2\right\} \leq m+n, \quad \text{for } 1 \leq i \leq m+n .$$

where we use that $0 < a_{ij} < 1$ and $\hat{c} \leq 1$.

The input of each following application of the LLL-algorithm is derived from the reduced basis found in the previous iteration by making some of the entries strictly smaller. Part (ii) of Proposition 6 yields that for every vector b_i in a reduced basis it holds that

$$|b_i|^2 \leq 2^{\frac{(m+n)(m+n-1)}{2}} (\det(L))^2 \prod_{j=1, j\neq i}^{m+n} |b_i|^{-2}.$$

The determinant of our starting lattice is given by $\hat{c}^m$ and the determinants of all subsequent lattices are strictly smaller. Every vector b_i in the lattice is at least as long as the shortest non-zero vector in the lattice. Thus for each i we have $|b_i|^2 \geq \frac{1}{2^M}$. Combining this yields

$$|b_i|^2 \leq 2^{\frac{(m+n+2M)(m+n-1)}{2}} \hat{c}^{2m} \leq 2^{\frac{(m+n+2M)(m+n-1)}{2}}$$

for every vector used as input for the LLL-step after the first iteration.

So we have

$$|b_i|^2 < \max\left\{m+n\,,\, 2^{\frac{(m+n+2M)(m+n-1)}{2}}\right\} = 2^{\frac{(m+n+2M)(m+n-1)}{2}} \tag{42}$$

for any basis vector that is used as input for an LLL-step in the ILLL-algorithm.

Proposition 7 shows that for a given basis $b_1, \ldots, b_{m+n}$ for $\mathbb{Z}^{m+n}$ with $F \in \mathbb{R}$, $F \geq 2$ such that $|b_i|^2 \leq F$ for $1 \leq i \leq m+n$ the number of arithmetic operations needed to find a reduced basis from this input is $O((m+n)^4 \log F)$. For matrices with entries in $\mathbb{Q}$ we need to clear denominators before applying this proposition. Thus for a basis with basis vectors $|b_i|^2 \leq F$ and rational entries that can all be written as fractions with denominator 2^M the number of arithmetic operations is $O((m+n)^4 \log(2^{2M} F))$.

Combining this with (42) and the number of iterations yields the theorem. □

4.2. **Approximation results from the rational algorithm.** Assume that the input matrix A (with entries $a_{ij} = \frac{p_{ij}}{2^M} \in \mathbb{Q}$) is an approximation of an $n \times m$-matrix $\mathcal{A}$ (with entries $\alpha_{ij} \in \mathbb{R}$), found by putting $a_{ij} = \frac{\lceil 2^M \alpha_{ij} \rfloor}{2^M}$. In this subsection we derive the approximation results guaranteed by the rational iterated algorithm for the $\alpha_{ij} \in \mathbb{R}$.

According to (12) and (13) the LLL-algorithm applied with $\hat{c}$ instead of c guarantees to find an m-tuple $(q_1, \ldots, q_m)$ such that

$$\begin{aligned}
q = \max_j |q_j| \quad &\leq \quad 2^{\frac{(m+n-1)(m+n)}{4m}} \varepsilon^{\frac{-n}{m}}, \\
&\text{and} \\
\max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| \quad &\leq \quad 2^{\frac{m+n-1}{4}} \left(\left(2^{-\frac{m+n-1}{4}} \varepsilon \right)^{\frac{m+n}{m}} + \frac{1}{2^M} \right)^{\frac{m}{m+n}} \\
&\leq \quad \varepsilon + 2^{\frac{(m+n-1)(m+n)-4Mm}{4(m+n)}};
\end{aligned}$$

the last inequality follows from $(x+y)^\alpha \leq x^\alpha + y^\alpha$ if $\alpha < 1$ and $x, y > 0$.

For the α_{ij} we find that

$$\begin{aligned}
\max_i \|q_1 \alpha_{i1} + \cdots + q_m \alpha_{im}\| &\leq \max_i \|q_1 a_{i1} + \cdots + q_m a_{im}\| + mq2^{-M} \\
&\leq \varepsilon + 2^{\frac{m+n-1}{4} - \frac{Mm}{m+n}} + m\varepsilon^{\frac{-n}{m}} 2^{\frac{(m+n-1)(m+n)}{4m} - M}.
\end{aligned}$$

In the introduction to Section 4 we have chosen M large enough to guarantee that the error introduced by rationalizing the entries is negligible.

We show that the difference between $\hat{c}(k)$ and $c(k)$ is bounded by $\frac{2}{2^M}$.

Lemma 43. *For each integer $k \geq 0$,*

$$c(k) \leq \hat{c}(k) < c(k) + \frac{1}{2^M} \sum_{i=0}^{k} 2^{-\frac{i(m+n)}{m}} < c(k) + \frac{2}{2^M}.$$

Proof. We use induction. For $k = 0$ we have $\hat{c}(0) = \frac{\lceil c(0) 2^M \rceil}{2^M}$ and trivially

$$c(0) \leq \hat{c}(0) < c(0) + \frac{1}{2^M}.$$

Assume that $c(k-1) \leq \hat{c}(k-1) < c(k-1) + \frac{1}{2^M} \sum_{i=0}^{k-1} 2^{-\frac{i(m+n)}{m}}$ and consider $\hat{c}(k)$. From the definition of $\hat{c}(k)$ and the induction assumption it follows that

$$\hat{c}(k) = \frac{\left\lceil \hat{c}(k-1)\, 2^{-\frac{m+n}{m}} 2^M \right\rceil}{2^M} \geq \frac{\hat{c}(k-1)}{2^{\frac{m+n}{m}}} \geq \frac{c(k-1)}{2^{\frac{m+n}{m}}} = c(k)$$

and

$$\begin{aligned}
\hat{c}(k) = \frac{\left\lceil \hat{c}(k-1)\, 2^{-\frac{m+n}{m}} 2^M \right\rceil}{2^M} &< \frac{\hat{c}(k-1)}{2^{\frac{m+n}{m}}} + \frac{1}{2^M} \\
&< \frac{c(k-1) + \frac{1}{2^M} \sum_{i=0}^{k-1} 2^{-\frac{i(m+n)}{m}}}{2^{\frac{m+n}{m}}} + \frac{1}{2^M} \\
&= c(k) + \frac{1}{2^M} \sum_{i=0}^{k} 2^{-\frac{i(m+n)}{m}}.
\end{aligned}$$

Finally note that $\sum_{i=0}^{k} 2^{-\frac{i(m+n)}{m}} < 2$ for all k. □

One can derive analogues of Theorem 24, Lemma 26 and Theorem 36 for the polynomial version of the ILLL-algorithm by carefully adjusting for the introduced error. We do not give the details, since in practice this error is negligible.

5. Experimental data

In this section we present some experimental data from the rational ILLL-algorithm. In our experiments we choose the dimensions m and n and iteration speed d, so $\varepsilon = \frac{1}{d}$. We fill the $m \times n$ matrix A with random numbers in the interval $[0, 1]$ and repeat the entire ILLL-algorithm for a large number of these random matrices to find our results. First we look at the distribution of the approximation quality. Then we look at the growth of the denominators q found by the algorithm.

5.1. **The distribution of the approximation qualities.** For one-dimensional continued fractions the approximation coefficients Θ_k are defined as

$$\Theta_k = q_k^2 \left| a - \frac{p_k}{q_k} \right|,$$

where p_k/q_k is the kth convergent of a.

For the multi-dimensional case we define Θ_k in a similar way

$$\Theta_k = q(k)^{\frac{m}{n}} \max_i \|q_1(k)\, a_{i1} + \cdots + q_m(k)\, a_{im}\|. \tag{44}$$

The one-dimensional case $m = n = 1$*.* We compare the distribution of the Θ_k's found by the ILLL-algorithm for $m = n = 1$ and various values of d with the distribution of the Θ_k's as produced by the continued fraction algorithm with the best approximation properties. For this optimal continued fraction algorithm it was shown in [2] that for almost all a

$$\lim_{N\to\infty} \frac{1}{N} \#\{1 \le k \le N : \Theta_k \le z\} = F(z),$$

where

$$F(z) = \begin{cases} \dfrac{z}{\log G}, & 0 \le z \le \frac{1}{\sqrt{5}}, \\[2ex] \dfrac{\sqrt{1-4z^2} + \log(G\frac{1-\sqrt{1-4z^2}}{2z})}{\log G}, & \frac{1}{\sqrt{5}} \le z \le \frac{1}{2}, \\[2ex] 1, & \frac{1}{2} \le z \le 1, \end{cases}$$

with $G = \frac{\sqrt{5}+1}{2}$.

The optimal continued fraction algorithm finds rational approximations of which the denominators grow with maximal rate, and it finds all approximations with $\Theta_k < \frac{1}{2}$; for all this, see [1, 2, 3].

The following figures display distribution functions for Θ_k, that is, we show the fraction of Θ_k's found up to the value given on the horizontal axis.

We plot the distribution of the Θ_k's found by the ILLL-algorithm for $m = n = 1$ and $d = 2$ in Figure 1. The ILLL-algorithm might find the same approximation more than once. We see in Figure 1 that for $d = 2$ the distribution function differs depending on whether we leave in the duplicates or sort them out. With the duplicate approximations removed the distribution of Θ_k strongly resembles $F(z)$ of the optimal continued fraction. The duplicates that the ILLL-algorithm finds are usually good approximations: if they are much better than necessary they will also be an admissible solution in the next few iterations.

For larger d we do not find so many duplicates, because the quality has to improve much more in every iteration; also see Figure 2 for an example with $d = 64$.

From now on we remove duplicates from our results.

5.2. **The multi-dimensional case.** In this section we show some results for the distribution of the Θ_k's found by the ILLL-algorithm. For fixed m and n there also appears to be a limit distribution for Θ_k as d grows. See Figure 4 for an example with $m = 3$ and $n = 2$, and compare this with Figure 3. In this section we fix $d = 512$.

In Figure 5 we show some distributions for cases where either m or n is 1.

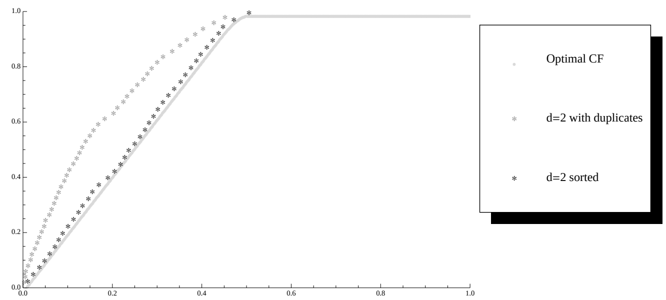

FIGURE 1. The distribution function for Θ_k from ILLL with $m = n = 1$ and $d = 2$, with and without the duplicate approximations, compared to the distribution function of Θ_k for optimal continued fractions.

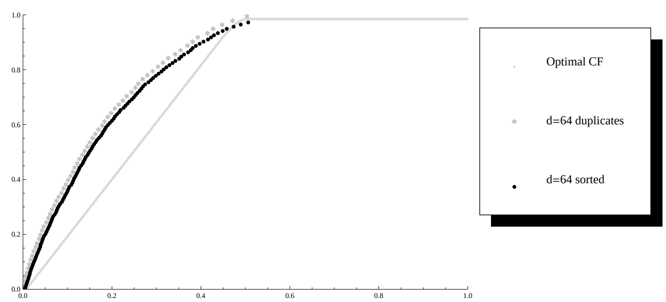

FIGURE 2. The distribution function for Θ_k from ILLL with $m = n = 1$ and $d = 64$, with and without the duplicate approximations, compared to the distribution function of Θ_k for optimal continued fractions.

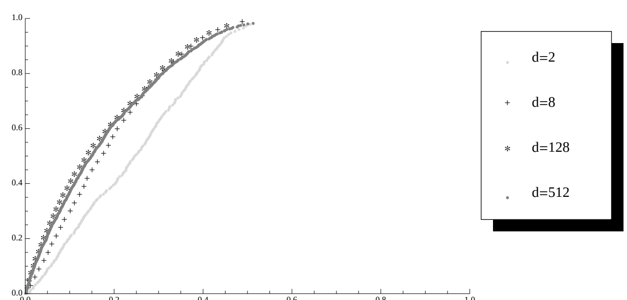

FIGURE 3. The distribution function for Θ_k from ILLL (with duplicates removed) with $m = n = 1$ and various values of d.

In Figure 6 we show some distributions for cases where $m = n$.

Remark 45. Very rarely the ILLL-algorithm returns an approximation with $\Theta_k > 1$.

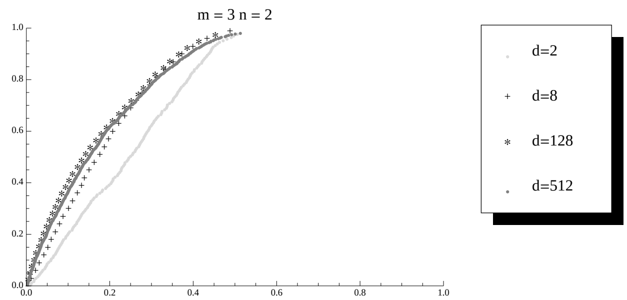

FIGURE 4. The distribution function for Θ_k from ILLL with $m = 3$ and $n = 2$ for $d = 2, 8, 128$ and 512.

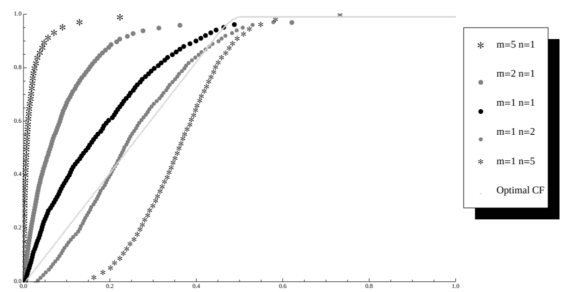

FIGURE 5. The distribution for Θ_k from ILLL when either $m = 1$ or $n = 1$.

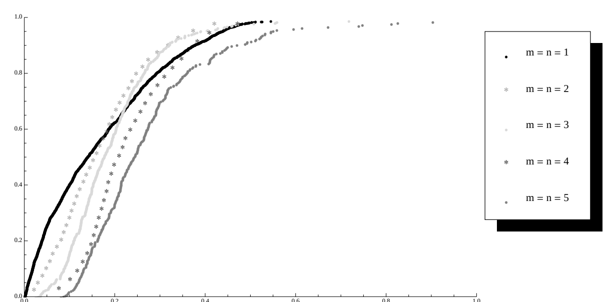

FIGURE 6. The distribution of Θ_k from ILLL when $m = n$.

5.3. **The denominators** q**.** For regular continued fractions, the denominators grow exponentially fast, to be more precise, for almost all x we have that

$$\lim_{k\to\infty} q_k^{1/k} = e^{\frac{\pi^2}{12\log 2}},$$

see Section 3.5 of [6].

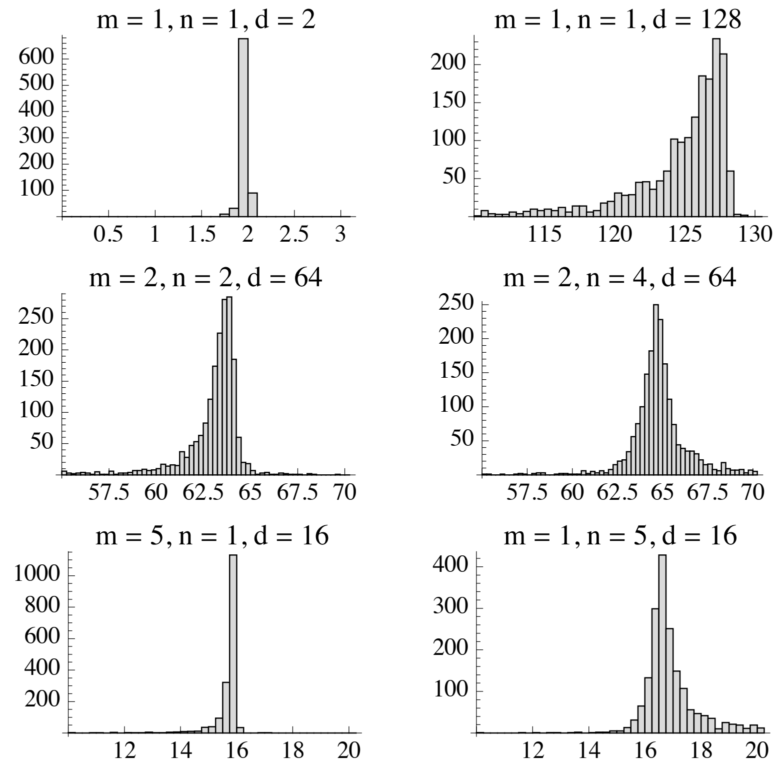

FIGURE 7. Histograms of $e^{\frac{m \log q(k)}{k\, n}}$ for various values of m, n and d. In these experiments we used $q_{\max} = 10^{40}$ and repeated the ILLL-algorithm $\left\lfloor \frac{2000}{k'} \right\rfloor$ times, with k' from Lemma 22.

For optimal continued fractions the constant $\frac{\pi^2}{12 \log 2}$ is replaced by $\frac{\pi^2}{12 \log G}$ with $G = \frac{\sqrt{5}+1}{2}$. For multi-dimensional continued fraction algorithms little is known about the distribution of the denominators q_j. Lagarias defined in [11] the notion of a best simultaneous Diophantine approximation and showed that for the ordered denominators $1 = q_1 < q_2 < \ldots$ of best approximations for $a_1, \ldots, a_n$ it holds that

$$\lim_{k \to \infty} \inf q_k^{1/k} \geq 1 + \frac{1}{2^{n+1}}.$$

We look at the growth of the denominators $q = \max_j |q_j|$ that are found by the ILLL-algorithm. Dirichlet's Theorem 2 suggests that if q grows exponentially with a rate of m/n, then infinitely many approximations with Dirichlet coefficient smaller than 1 can be found. In the iterated LLL-algorithm it is guaranteed by (16) that $q(k)$ is smaller than a constant times $d^{\frac{kn}{m}}$. Our experiments indicate that $q(k)$ is about $d^{\frac{kn}{m}}$, or equivalently that $e^{\frac{m \log q_k}{k\, n}}$ is about d; see Figure 7 which gives a histogram of solutions that satisfy $e^{\frac{m \log q_k}{k\, n}} = x$.

References

[1] W. Bosma. Optimal continued fractions. *Indag. Math.*, 49: 353–379, 1987.

[2] W. Bosma and C. Kraaikamp. Metrical theory for optimal continued fractions. *J. Number Theory*, 34(3): 251–270, 1990.

[3] W. Bosma and C. Kraaikamp. Optimal approximation by continued fractions. *J. Austral. Math. Soc.* (Series A), 50: 481–504, 1991.

[4] A. J. Brentjes. *Multidimensional continued fraction algorithms*, volume 145 of *Mathematical Centre Tracts*. Mathematisch Centrum, Amsterdam, 1981.

[5] V. Brun. En generalisation av kjedebroken i+ii. *Skr. Vid. Selsk. Kristiana, Mat. Nat. 6 (1919) and 6 (1920)*, 1919.

[6] K. Dajani and C. Kraaikamp. *Ergodic theory of numbers*, volume 29 of *Carus Mathematical Monographs*. Mathematical Association of America, Washington, DC, 2002.

[7] H. R. P. Ferguson and R. W. Forcade. Generalization of the Euclidean algorithm for real numbers to all dimensions higher than two. *Bull. Amer. Math. Soc. (N.S.)*, 1(6): 912–914, 1979.

[8] A. Hurwitz. Über die angenäherte Darstellung der Zahlen durch rationale Brüche. *Math. Ann.*, 44(2-3): 417–436, 1894.

[9] C. Jacobi. Allgemeine Theorie der kettenbruchähnlichen Algorithmen. *J. reine angew. Math.*, 69: 29–64, 1868.

[10] B. Just. Generalizing the continued fraction algorithm to arbitrary dimensions. *SIAM J. Comput.*, 21(5): 909–926, 1992.

[11] J. C. Lagarias. Best simultaneously diophantine approximations. i. growth rates of best approximation denominators. *Trans. Amer. Math. Soc.*, 272(2): 545–554, 1982.

[12] J. C. Lagarias. The computational complexity of simultaneous Diophantine approximation problems. *SIAM J. Comput.*, 14(1): 196–209, 1985.

[13] J. C. Lagarias. Geodesic multidimensional continued fractions. *Proc. London Math. Soc. (3)*, 69(3): 464–488, 1994.

[14] A. M. Legendre. Essai sur la théorie des nombres, 1798.

[15] G. Lejeune Dirichlet. *Mathematische Werke. Bände I, II.* Herausgegeben auf Veranlassung der Königlich Preussischen Akademie der Wissenschaften von L. Kronecker. Chelsea Publishing Co., Bronx, N.Y., 1969.

[16] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4): 515–534, 1982.

[17] O. Perron. Grundlagen für eine Theorie des Jacobischen Kettenbruchalgorithmus. *Math. Ann.*, 64(1): 1–76, 1907.

[18] W. M. Schmidt. *Diophantine approximation*, volume 785 of *Lecture Notes in Mathematics*. Springer, Berlin, 1980.

[19] F. Schweiger. *Multidimensional continued fractions*. Oxford Science Publications. Oxford University Press, Oxford, 2000.

[20] Ionica Smeets. *On continued fraction algorithms*. Ph.D. thesis. Leiden, 2010. Online available at `www.math.leidenuniv.nl/scripties/SmeetsThesis.pdf`

Wieb Bosma, Mathematics, Radboud University Nijmegen, PO Box 9010, 6500 GL Nijmegen, the Netherlands

E-mail address: `bosma@math.ru.nl`

Ionica Smeets, Mathematical Institute, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, the Netherlands

E-mail address: `ionica.smeets@gmail.com`