Acta Univ. Sapientiae, Informatica, **5**, 1 (2013) 16–34

Efficient computing of n-dimensional simultaneous Diophantine approximation problems

Attila KOVÁCS
Eötvös Loránd University
Faculty of Informatics
email:
attila.kovacs@compalg.inf.elte.hu

Norbert TIHANYI
Eötvös Loránd University
Faculty of Informatics
email:
ntihanyi@compalg.inf.elte.hu

Abstract. In this paper we consider two algorithmic problems of simultaneous Diophantine approximations. The first algorithm produces a full solution set for approximating an irrational number with rationals with common denominators from a given interval. The second one aims at finding as many simultaneous solutions as possible in a given time unit. All the presented algorithms are implemented, tested and the PariGP version made publicly available.

1 Introduction

1.1 The problem statement

Rational approximation, or alternatively, Diophantine approximation is very important in many fields of mathematics and computer science. Archimedes approximated the irrational number π with $22/7$. Long before Archimedes, ancient astronomers in Egypt, Babylonia, India and China used rational approximations. While the work of John Wallis (1616–1703) and Christiaan Huygens (1629–1695) established the field of continued fractions, it began to blossom

Computing Classification System 1998: G.2.0
Mathematics Subject Classification 2010: 68R01, 11J68
Key words and phrases: Diophantine approximation

when Leonhard Euler (1707–1783), Johann Heinrich Lambert (1728–1777) and Joseph Louis Lagrange (1736–1813) embraced the topic. In the 1840s, Joseph Liouville (1809–1882) obtained an important result on general algebraic numbers: if α is an irrational algebraic number of degree $n > 0$ over the rational numbers, then there exists a constant $c(\alpha) > 0$ such that

$$\left|\alpha - \frac{p}{q}\right| > \frac{c(\alpha)}{q^n}$$

holds for all integers p and $q > 0$. This result allowed him to produce the first proven examples of transcendental numbers. In 1891 Adolf Hurwitz (1859–1919) proved that for each irrational α infinitely many pairs (p, q) of integers satisfy

$$\left|\alpha - \frac{p}{q}\right| < \frac{1}{q^2\sqrt{5}}\,,$$

but there are some irrational numbers β for which at most finitely many pairs satisfy

$$\left|\beta - \frac{p}{q}\right| < \frac{1}{q^{2+\gamma}\sqrt{5+\mu}}$$

no matter how small the positive increments γ and μ are.

The idea can be generalized to simultaneous approximation. Simultaneous diophantine approximation originally means that for given real numbers $\alpha_1, \alpha_2, \ldots, \alpha_n$ find $p_1, p_2, \ldots, p_n, q \in \mathbb{Z}$ such that

$$\left|\alpha_i - \frac{p_i}{q}\right|$$

is "small" for all i, and q is "not too large".

For a given real α let us denote the nearest integer distance function by $\|.\|$, that is, $\|\alpha\| = \min\{|\alpha - j|, j \in \mathbb{Z}\}$. Then, simultaneous approximation can be interpreted as minimizing

$$\max\{\|q\alpha_1\|, \ldots, \|q\alpha_n\|\}\,.$$

In 1842 Peter Gustav L. Dirichlet (1805–1859) showed that there exist simultaneous Diophantine approximations with absolute error bound $q^{-(1+1/n)}$. To be more precise, he showed that there are infinitely many approximations satisfying

$$|q \cdot \alpha_i - p_i| < \frac{1}{q^{1/n}} \tag{1}$$

for all $1 \leq i \leq n$. Unfortunately, no polynomial algorithm is known for the simultaneous Diophantine approximation problem. However, due to the L^3 algorithm of Lenstra, Lenstra and Lovász, if $\alpha_1, \alpha_2, \ldots, \alpha_n$ are irrationals and $0 < \varepsilon < 1$ then there is a polynomial time algorithm to compute integers $p_1, p_2, \ldots, p_n, q \in \mathbb{Z}$ such that

$$1 \leq q \leq 2^{n(n+1)/4}\varepsilon^{-n} \text{ and } |q \cdot \alpha_i - p_i| < \varepsilon$$

for all $1 \leq i \leq n$ (see [10]).

Lagarias [7, 8] presented many results concerning the best simultaneous approximations. Szekeres and T. Sós [12] analyzed the signatures of the best approximation vectors. Kim et al. [4] discussed rational approximations to pairs of irrational numbers which are linearly independent over the rationals and applications to the theory of dynamical systems. Armknecht et al. [1] used the inhomogeneous simultaneous approximation problem for designing cryptographic schemes. Lagarias [9] discussed the computational complexity of Diophantine approximation problems, which, depending on the specification, varies from polynomial-time to $\mathcal{NP}$-complete. Frank and Tardos [2] developed a general method in combinatorial optimization using simultaneous Diophantine approximations which could transform some polynomial time algorithms into strongly polynomial.

In this paper we focus on two algorithmic problems. Consider the set of irrationals $\Upsilon = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$. Let $\varepsilon > 0$ be real and $1 \leq a \leq b$ be natural numbers. Furthermore, let us define the set

$$\Omega(\Upsilon, \varepsilon, a, b) = \{k \in \mathbb{N} : a \leq k \leq b, \|k\alpha_i\| < \varepsilon \text{ for all } \alpha_i \in \Upsilon\}. \tag{2}$$

For given Υ, ε and a, b

1. determine all the elements of $\Omega(\Upsilon, \varepsilon, a, b)$,
2. determine as many elements of $\Omega(\Upsilon, \varepsilon, a, b)$ as possible in a given time unit

efficiently. We refer to the first problem as the "all-elements simultaneous Diophantine approximation problem". In case of $|\Upsilon| = n \geq 1$ we call it an n-dimensional simultaneous approximation. The second problem is referred to as the "approximating as many elements as possible" problem.

Challenges:

1. Determine all elements of

$$\Omega\big(\{\sqrt{2}\}, 10^{-17}, 10^{20}, 10^{21}\big). \tag{3}$$

2. Determine as many elements of

$$\Omega\left(\left\{\frac{\log(p)}{\log(2)}\,,p \text{ prime}\,,3 \le p \le 19\right\},10^{-2},1,10^{18}\right) \tag{4}$$

as possible in a given time unit.

1.2 The continued fraction approach

It is well-known that continued fractions are one of the most effective tools of rational approximation to a real number. *Simple continued fractions* are expressions of the form

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cdots}}$$

where a_i are integer numbers with $a_1, a_2, \ldots > 0$. It is called *finite* if it terminates, and *infinite* otherwise. These continued fractions are usually represented in bracket form $[a_0, a_1, \ldots, a_m, \ldots]$, i.e.

$$C_0 = [a_0] = a_0, \quad C_1 = [a_0, a_1] = a_0 + \frac{1}{a_1}, \quad C_2 = [a_0, a_1, a_2] = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2}}, \quad \ldots$$

where C_m are called *convergents*. Clearly, the convergents C_m represent some rational numbers p_m/q_m. An infinite continued fraction $[a_0, a_1, a_2, \ldots]$ is called convergent if its sequence of convergents C_m converges in the usual sense, i.e. the limit

$$\alpha = \lim_{m\to\infty} C_m = \lim_{m\to\infty} [a_0, a_1, \ldots, a_m]$$

exists. In this case we say that the continued fraction represents the real number α. The simple continued fraction expansion of $\alpha \in \mathbb{R}$ is infinite if and only if α is irrational. The convergents C_m are the best rational approximations in the following sense:

Lemma 1 *No better rational approximation exists to the irrational number α with smaller denominator than the convergents $C_m = p_m/q_m$.*

Example 2 *The simple continued fraction approximation for $\sqrt{2}$ is $[1, 2, 2, \ldots]$, the sequence of the convergents is*

$$1, \frac{3}{2}, \frac{7}{5}, \frac{17}{12}, \frac{41}{29}, \frac{99}{70}, \frac{239}{169}, \frac{577}{408}, \frac{1393}{985}, \frac{3363}{2378}, \frac{8119}{5741}, \ldots$$

Among all fractions with denominator at most 29*, the fraction* $41/29$ *is the closest to* $\sqrt{2}$*, among all fractions with denominator at most* 70*, the fraction* $99/70$ *is closest to* $\sqrt{2}$*, and so on.*

Every convergent is a best rational approximation, but these are not all of the best rational approximations. Fractions of the form

$$\frac{p_{m-1} + jp_m}{q_{m-1} + jq_m} \ (1 \leq j \leq a_{m+2} - 1),$$

are called *intermediate convergents* or *semi-convergents*. To get every rational approximation between two consecutive p_m/q_m and p_{m+1}/q_{m+1}, we have to calculate the intermediate convergents.

Example 2 (cont.) *The missing intermediate convergents of Example* 2 *are*

$$\frac{4}{3}, \frac{10}{7}, \frac{24}{17}, \frac{58}{41}, \frac{140}{99}, \frac{338}{239}, \frac{816}{577}, \frac{1970}{1393}, \frac{4756}{3363}, \ldots$$

The approximations $|\alpha - p/q|$ above are also known as "best rational approximations of the first kind". However, sometimes we are interested in the approximations $|\alpha \cdot q - p|$. This is called the *approximation of a second kind.*

Lemma 3 [3] *A rational number* p/q*, which is not an integer, is a convergent of a real number* α *if and only if it is a best approximation of the second kind of* α*.*

In 1997 Clark Kimberling proved the following result regarding intermediate convergents [5]:

Lemma 4 *The best lower (upper) approximates to a positive irrational number* α *are the even-indexed (odd-indexed) intermediate convergents.*

Example 2 (cont.) *In order to generate many integers* q *that satisfy*

$$\|q \cdot \sqrt{2}\| < 10^{-5} \tag{5}$$

one can apply the theory of continued fractions, especially convergents. If q_m *is the first integer that satisfies* $\|q_m \cdot \sqrt{2}\| < 10^{-5}$ *in the continued fraction expansion of* $\sqrt{2}$*, then all convergents with denominator larger than* q_m *will satisfy equation* (5)*.*

Example 5 *Consider Challenge* 1 *stated in* (3). *There are only* 3 *convergents of* $\sqrt{2}$ *where* $10^{20} < q_m < 10^{21}$. *They are*

$$\frac{23380673249993320809 9}{16532632603777192063 0}, \quad \frac{56445938457547704935 9}{39913305853770512872 9}, \quad \frac{13627255016508873068 17}{96359244311318217808 8}.$$

With intermediate convergents we get 2 *more solutions. Hence, with the theory of continued fractions we are able to find only* 5 *appropriate integers. One may ask how many elements are in the set* Ω *in* (3)*?*

Hermann Weyl (1855–1955) and Waclaw Sierpiński (1882–1969) proved in 1910 that if $\alpha \in \mathbb{R} \setminus \mathbb{Q}$ then $\alpha, 2\alpha, 3\alpha, \ldots \pmod 1$ is uniformly distributed on the unit interval. From this theorem it immediately follows that there are approximately $2(b-a)\varepsilon$ appropriate integers in the $[a, b]$ interval. In Challenge 1 we expect $2(10^{21} - 10^{20}) \cdot 10^{-17} = 18000$ (± 1) integers. This is by several orders of magnitude more than what we were able to obtain by continued fractions.

1.3 The Lenstra–Lenstra–Lovász approach

We have seen in the previous section that Challenge 1 is unsolvable with the theory of continued fractions. Challenge 2 is a 7-dimensional simultaneous approximation problem and is even more beyond the potentials of continued fractions. Although there is not known polynomial-time algorithm that is able to solve the Dirichlet type simultaneous Diophantine approximation problem, there exists an algorithm that can be useful for *similar* problems. The Lenstra–Lenstra–Lovász basis reduction algorithm (L^3) is a polynomial-time algorithm that finds a reduced basis in a lattice [10]. The algorithm can be applied to solve simultaneous Diophantine approximation *with an extra condition.*

Lemma 6 *There exists a polynomial-time algorithm for the given irrationals* $\alpha_1, \alpha_2, \ldots, \alpha_n$ *and* $0 < \varepsilon < 1$ *that can compute the integers* $p_1, \ldots, p_n$ *and* q *such that*

$$\left|\alpha_i - \frac{p_i}{q}\right| < \frac{\varepsilon}{q} \tag{6}$$

and

$$0 < q \leq \beta^{n(n+1)/4} \varepsilon^{-n}$$

hold for all $1 \leq i \leq n$, *where* β *is an appropriate reduction parameter.*

The extra condition is the bound $0 < q \leq \beta^{n(n+1)/4} \varepsilon^{-n}$.

In one-dimension the L^3 algorithm provides exactly the continued fraction approach discussed in the previous section, hence L^3 is not an effective tool for answering Challenge 1. And what about the multidimensional case like Challenge 2?

Let $\alpha_1, \alpha_2, \ldots, \alpha_n$ be irrational numbers and let us approximate them with rationals admitting an $\varepsilon > 0$ error. Let $X = \beta^{n(n+1)/4}\varepsilon^{-n}$ and let the matrix A be the following:

$$A = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ \alpha_1 X & X & 0 & \ldots & 0 \\ \alpha_2 X & 0 & X & \ldots & 0 \\ \vdots & & & & \vdots \\ \\ \alpha_n X & 0 & 0 & \ldots & X \end{bmatrix}.$$

Applying the L^3 algorithm for A, the first column of the resulting matrix contains the vector $[q, p_1, p_2, p_3, \ldots, p_n]^T$ which satisfies (6).

Let us see how the L^3 algorithm works in dimension 7. Let $\alpha_i = \frac{\log(p_{i+1})}{\log(2)}$ where p_i denotes the i-th prime for $1 \le i \le 7$, and let $\varepsilon = 0.01$. We are looking for an integer $q \le 2^{14} \cdot 100^7$ that satisfies $\|q \cdot \alpha_i\| < \varepsilon$ for all i. Applying the L^3 algorithm we got $q = 1325886000944418$. It is easy to verify that $\|q\, \alpha_i\| < 0.01$ holds for all $1 \le i \le 7$.

The L^3 algorithm can also be applied in higher dimensions, however, there are some cases where the algorithm can not be used efficiently. The real drawback of the method for our purposes is that it is inappropriate for finding all or many different solutions q in an *arbitrary interval.* We note that sometimes one can find a few more solutions with a different choice of β (but not much more).

It can be concluded that the apparatus of the continued fractions and the L^3 algorithm is not appropriate for solving Challenge 1 and Challenge 2 problems. In this paper we present new methods that can be used to solve these kinds of problems efficiently. All the algorithms presented in this paper were implemented and tested in PARI/GP 2.5.3 with an extension of GNU MP 5.0.1. The experimenting environment was an Intel® Core i5-2450M with Sandy Bridge architecture. The code can be downloaded from the project homepage[1].

[1] http://www.riemann-siegel.com/

2 Approximation in the one-dimensional case

2.1 "All-elements" approximation

In this section we present how to calculate all the elements of $\Omega(\Upsilon, \varepsilon, a, b)$ where $\Upsilon = \{\alpha\}$.

For a given Ω let $k : \{1, 2, \dots, |\Omega|\} \to \Omega$ monotonically increasing, so k_i denotes the ith integer in Ω. Let us define the set

$$\Delta_\Omega = \{k_{n+1} - k_n : \ 1 \le n \le |\Omega| - 1\}.$$

The set Δ_Ω contains all possible step-sizes between two consecutive k_i's.

Theorem 7 $|\Delta_\Omega| \le 3$.

Proof. The proof has two parts. In the first step we construct all the possible three elements of Δ_Ω and in the second step we show that there is no more. For the given irrational α and an arbitrary $m \in \mathbb{N}$ let

$$\langle m \rangle = \begin{cases} \|\alpha m\| & \text{if } \alpha m - \|\alpha m\| \in \mathbb{N}\,, \\ -\|\alpha m\| & \text{if } \alpha m + \|\alpha m\| \in \mathbb{N}\,. \end{cases}$$

Let us furthermore define the following open intervals:

$$A = (-2\varepsilon, -\varepsilon), \ B = (-\varepsilon, 0), \ C = (0, \varepsilon), \ D = (\varepsilon, 2\varepsilon)\,. \tag{7}$$

Let m_1 be the smallest positive integer that satisfies $\langle m_1 \rangle \in C \cup D$, let m_2 be the the smallest positive integer that satisfies $\langle m_2 \rangle \in A \cup B$ and let $m_3 = m_1 + m_2$.

The first part of the proof is to show that there is always at least one integer $(m_1, m_2$ or $m_3)$ that adding to an arbitrary $k_i \in \Omega$ always produces a new integer $k_j \in \Omega$. Clearly, $\langle k_i \rangle \in B \cup C$ for all k_i. Let us see the following cases:

$\langle k_i \rangle \in B$:

If $\langle m_1 \rangle \in C$, $\langle m_2 \rangle \in A \cup B$ then $\langle k_i + m_1 \rangle \in B \cup C$.

If $\langle m_1 \rangle \in D$, $\langle m_2 \rangle \in A$ and $\langle m_1 + m_2 \rangle \in C$ then $\langle k_i + (m_1 + m_2) \rangle \in B \cup C$.

If $\langle m_1 \rangle \in D$, $\langle m_2 \rangle \in A$ and $\langle m_1 + m_2 \rangle \in B$ then $\langle k_i + (m_1 + m_2) \rangle \in A \cup B$. If $\langle k_i + (m_1 + m_2) \rangle \in A$ then $\langle k_i + (m_1 + m_2) - m_2 \rangle \in B \cup C$.

If $\langle m_1 \rangle \in D$, $\langle m_2 \rangle \in B$ and $\langle m_1 + m_2 \rangle \in C$ then $\langle k_i + (m_1 + m_2) \rangle \in B \cup C$.

If $\langle m_1 \rangle \in D$, $\langle m_2 \rangle \in B$ and $\langle m_1 + m_2 \rangle \in D$ then $\langle k_i + (m_1 + m_2) \rangle \in C \cup D$. If $\langle k_i + (m_1 + m_2) \rangle \in D$ then $\langle k_i + (m_1 + m_2) - m_1 \rangle \in B \cup C$.

$\langle k_i \rangle \in C$:

If $\langle m_1 \rangle \in C \cup D$, $\langle m_2 \rangle \in B$ then $\langle k_i + m_2 \rangle \in B \cup C$.

If $\langle m_1 \rangle \in C$, $\langle m_2 \rangle \in A$ and $\langle m_1 + m_2 \rangle \in B$ then $\langle k_i + (m_1 + m_2) \rangle \in B \cup C$.

If $\langle m_1 \rangle \in C$, $\langle m_2 \rangle \in A$ and $\langle m_1 + m_2 \rangle \in A$ then $\langle k_i + (m_1 + m_2) \rangle \in A \cup B$.
If $\langle k_i + (m_1 + m_2) \rangle \in A$ then $\langle k_i + (m_1 + m_2) - m_2 \rangle \in B \cup C$.

If $\langle m_1 \rangle \in D$, $\langle m_2 \rangle \in A$ and $\langle m_1 + m_2 \rangle \in B$ then $\langle k_i + (m_1 + m_2) \rangle \in B \cup C$.

If $\langle m_1 \rangle \in D$, $\langle m_2 \rangle \in A$ and $\langle m_1 + m_2 \rangle \in C$ then $\langle k_i + (m_1 + m_2) \rangle \in C \cup D$.
If $\langle k_i + (m_1 + m_2) \rangle \in D$ then $\langle k_i + (m_1 + m_2) - m_1 \rangle \in B \cup C$.

Let now $X = \Delta_\Omega \setminus \{m_1, m_2, m_3\}$. We claim that $X = \emptyset$. Suppose otherwise, and let j be the smallest index with $m = k_{j+1} - k_j \in X$. Clearly, $\langle m \rangle \in A \cup B \cup C \cup D$. We can observe as well that for all $m \in \mathbb{N}$, $k_i \in \Omega$, $\langle k_i + m \rangle \in B \cup C$ implies $\langle m \rangle \in A \cup B \cup C \cup D$. Then it is easy to see that

- $j > 1$, and k_i's are integer linear combinations of m_1 and m_2 for all $i \leq j$,
- $m_1, m_2 < m < m_1 + m_2$,
- $\langle m \rangle \in A \cup D$.

If $\langle m \rangle \in A$ then $\langle m - m_2 \rangle \in B \cup C$, which contradicts the mimimality of j. In the same way, if $\langle m \rangle \in D$ then $\langle m - m_1 \rangle \in B \cup C$, which is a contradiction again. Hence, such an m does not exist. The proof is complete. □

Finding the integers m_1 and m_2 can be done very effectively with the theory of intermediate convergents. It was already discussed that intermediate convergents of an irrational α always produce the best upper and lower approximations to α, so m_1 and m_2 must be intermediate convergents.

Example 5 (cont.) *Applying the **FindMMM** algorithm (Algorithm 1) we have the values*
$m_1 = 59341817924539925$,
$m_2 = 24580185800219268$,
$m_3 = 83922003724759193$.
After the precalculation of m_1 *and* m_2 *it is very easy to compute every* k_i *between* 10^{20} *and* 10^{21}. *First we have to find an intermediate convergent between* 10^{20} *and* 10^{21}. *It can be done in polynomial time with the theory of continued fractions (e.g:* 233806732499933208099*). After that we can add, subtract* m_1, m_2 *or* m_3 *until we reach the bounds of the interval. The Weyl equidistribution theorem predicts* 18000 *integers that solve (3). Applying **Challenge 1 Solver***

algorithm (Algorithm 2) we found exactly 18 000 *integers. The precalculation and the computation of all* k_i *values took only* 31 *ms.*

Algorithm 1 FindMMM

Description:

The algorithm is based on Theorem 7. The algorithm finds the smallest m_1, m_2 and m_3 integers such that $0 < \langle m_1 \rangle < 2\varepsilon$, $-2\varepsilon < \langle m_2 \rangle < 0$. The output of the algorithm is $\Delta_\Omega = \{m_1, m_2, m_1 + m_2\}$. The main **while** loop in this algorithm (from line 5 to 15) goes through all intermediate convergents to find m_1 and m_2. The theory of intermediate convergents ensures that $m_1, m_2 \in q_i$ where q_i is the i^{th} intermediate convergent. When m_1 and m_2 are found the **while** loop terminates and the algorithm returns m_1,m_2 and $m_1 + m_2$ in ascending order.

Precondition: $\alpha \in \mathbb{R} \setminus \mathbb{Q}, \alpha > \varepsilon > 0.$

```
1:  procedure FindMMM(α, ε)
2:      i ← 0
3:      m1 ← 0
4:      m2 ← 0
5:      while m1 = 0 or m2 = 0  do
6:          i ← i + 1
7:          qi ← i^th intermediate convergents of α
8:          k ← Frac(qi · α)                    ▷ Fractional part of qi · α
9:          if m1 = 0 and k < 2ε then
10:             m1 ← qi
11:         end if
12:         if m2 = 0 and k > 1 − 2ε then
13:             m2 ← qi
14:         end if
15:     end while
16:     Return(min(m1, m2), max(m1, m2), m1 + m2)
17: end procedure
```

Algorithm 2 Challenge 1 Solver

Description:
The algorithm solves Challenge 1 (see (3)). Line 5 calls the FindMMM algorithm to determine Δ_Ω. With the theory of continued fractions line 6 finds an integer $k \in \Omega$. In the first **while** loop (lines $9-18$) the appropriate m_i is subtracted from k to generate a new integer $k_i \in \Omega$. The process is repeated until the lower bound A of the interval is reached. In the second **while** loop (lines $20-29$) the appropriate m_i is added to k generating $k_i \in \Omega$. The process is repeated until the upper bound of the interval B is reached. This method produces all the $18\,000$ integers that satisfy Challenge 1.

```
 1: x ← √2
 2: ε ← 10^-17
 3: A ← 10^20
 4: B ← 10^21
 5: v ← FindMMM(x, ε)
 6: k ← Find q_x in the interval [A,B] where FRAC(q_x · x) < ε
 7: ktemp ← k
 8: PRINT(k)
 9: while k > A do
10:     for i = 1 → 3 do
11:         ok ← FRAC((k − v[i]) · x)
12:         if (ok < ε) or (ok > 1 − ε) then k ← k − v[i]
13:             if k > A then PRINT(k)
14:             end if
15:             break                                  ▷ Leave the for loop
16:         end if
17:     end for
18: end while
19: k ← ktemp
20: while k < B do
21:     for i = 1 → 3 do
22:         ok ← FRAC((k + v[i]) · x)
23:         if (ok < ε) or (ok > 1 − ε) then k ← k + v[i]
24:             if k < B then PRINT(k)
25:             end if
26:             break                                  ▷ Leave the for loop
27:         end if
28:     end for
29: end while
```

2.2 "Many elements" approximation

In some cases it is not necessary to find all the k_i elements of Ω, rather it is enough to find as much as possible within a given time unit. Then, the following procedure works:

Find the smallest integer x that satisfies $0 < \langle x \rangle < \varepsilon$ and find the smallest integer y that satisfies $-\varepsilon < \langle y \rangle < 0$. Using the notations (7) it is easy to see that if $\langle k_i \rangle \in B$ and $\langle x \rangle \in C$ then $\langle k_i + x \rangle \in B \cup C$. In the same way, if $\langle k_i \rangle \in C$ and $\langle y \rangle \in B$ then $\langle k_i + y \rangle \in B \cup C$. Only with these two integers it is always possible to produce a subset of Ω.

Example 5 (cont.) *If we want to determine just "many" elements of* Ω, *the previous method generates* 12945 *integers within* 15 *ms.*

3 Approximations for the multi-dimensional case

3.1 "Many elements" approximation

Calculating all-elements of Ω seems to be hard in higher dimensions. However, we can generalize our one dimensional method to find "many" $q \in \Omega$ integers recursively. The algorithm is based on the following lemma:

Lemma 8 *Let the irrationals* $\alpha_1, \alpha_2, \dots, \alpha_n$ *and the real* $\varepsilon > 0$ *be given. Then there is a set* Γ_n *with* 2^n *elements with the following property: if* $q \in \Omega$ *then* $q + \gamma \in \Omega$ *for some* $\gamma \in \Gamma_n$.

Proof. Let $q \in \Omega$ be given. Let us define an n-dimensional binary vector b associated with q in the following way:

$$b_i = \begin{cases} 1 & \text{if } q\alpha_i - \|q\alpha_i\| \in \mathbb{N}, \\ 0 & \text{if } q\alpha_i + \|q\alpha_i\| \in \mathbb{N}\ . \end{cases} \tag{8}$$

Let Γ_n be the set for which

1. $\gamma \in \Gamma_n$ implies $\|q\alpha_i\| < \varepsilon$ for all $1 \leq i \leq n$,

2. all the associated binary representations by (8) are different.

Then, for a given $q \in \Omega$ there exists a $\gamma \in \Gamma_n$ such that $q + \gamma \in \Omega$, e.g. when their associated binary representations are (1's binary) complements. Clearly, $|\Gamma_n| = 2^n$. The proof is finished. $\square$

356205059916	3487229338057	3565485794412	3921690854328	4576624903864
5800642344603	7056176493393	7134432949748	7490638009664	9054007777845
10977867347721	11591199235356	11889764427290	12324225943561	15811455281618
16900850847425	17257055907341	18046611831809	18152923635291	18647375728749
18725632185104	19081837245020	19380402436954	19814863953225	20686645377416
20960788735295	21721870790627	22050020503416	22945888231366	23302093291282
23957027340818	25181044781557	25537249841473	25822432016319	27522513135230
27878718195146	28790615274715	29102735635885	29703499532825	31938492285330
32712306129043	35925048224463	38160204774654	39113548572900	39113712370586
40202944138707	41949469020031	42383930536302	42438100688898	44262882026577
44423200184969	44619087086493	44917652278427	47693582148371	51437938314147
51794143374063	52092708565997	56669333469861	58494114807540	59583346575661
60430542368111	62415805661868	62714370853802	63070575913718	65007167271975
65305732463909	65383988920264	66992266768046	67564975317859	68559097897151
68871218258321	75394965654965	76540726639349	76718061327901	76975188155620
79615221701227	79971426761143	81850378251418	82152413158738	84031364649013
84387569708929	87953055503341	88607825755191	88686082211546	91522002658677
91562625996499	92173311549603	95131573151835	95443693513005	96098463764855
98618802489892	98697058946247	100932215496438	103273520052425	104419444834495
105152471542700	107689663000211	112821979923728	119085139131729	121320131884234
140045764069338	140401969129254	143970916284590	147101940562731	151379836476975
153024924062435	156512153400492	161661323056483	164002791410156	170838495370284
175415120274148	179814246691774	183383193847110	189974502767900	204205735548863
208621878496649	208998700144938	261026707423816	266621541210731	269101092800260
269457297860176	299828135635546	305300628267360	320949406326919	331272339625104
382947603204990	408250989141648	616256074389738		

Table 1: The result of the precalculation for solving Challenge 2

Remark 9 *Computing the appropriate* $\gamma \in \Gamma_n$ *for a given* $q \in \Omega$ *is not necessarily unique.*

Corollary 10 *Remember the first dimension case: For all* $m \in \mathbb{N}$, $q \in \Omega$, $\langle q + m \rangle \in B \cup C$ *implies that* $\langle m \rangle \in A \cup B \cup C \cup D$. *We can generalize this to higher dimensions. Let* $q \in \Omega$ *and* $m \in \mathbb{N}$ *be given. Then* $q + m \in \Omega$ *implies* $\|m \cdot \alpha_i\| \in A \cup B \cup C \cup D$ *for all* $1 \leq i \leq n$.

Unfortunately, the precalculation of the 2^n integers is in general computationally expensive. However, there are several tricks based upon Lemma 8 that can be applied to make the generation more efficient.

Example 5 (cont.) *In Challenge* 2 *the precalculation of the* $2^7 = 128$ *integers took approximately* 6.14 *sec on our architecture. Table 1 shows the result. Applying the* ***Challenge*** 2 ***Solver*** *we were able to produce* 120852 *integers in* Ω *within* 26.8 *sec.*

Algorithm 3 **Challange 2 Solver**

Description:

The algorithm answers Challenge 2 (see (4)). Line 5 calls the PRECALC algorithm in order to determine the 2^n integers. The **while** loop generates a new integer in Ω using the precalculated ones. The method produces 120 852 integers that satisfy Challenge 2.

```
1: n ← 7
2: X ← log(p)/log(2) , p prime , 3 ≤ p ≤ 19
3: ε ← 0.01
4: B ← 10^18
5: v ← PRECALC(n, ε, X, 2^12)
6: k ← 0
7: while k < B do
8:     for i = 1 → length(v) do
9:         t ← TRUE
10:        for j = 1 → n do
11:            ok ← FRAC((k + v[i]) · X[j])
12:            if (ok > ε) and (ok < 1 − ε) then
13:                t ← FALSE
14:                break                                ▷ Leave the for loop
15:            end if
16:        end for
17:        if t = TRUE then
18:            k ← k + v[i]
19:            if k < B then
20:                PRINT(k)
21:            end if
22:            break
23:        end if
24:    end for
25: end while
```

Algorithm 4 Reduce

Description: The algorithm reduces the generation time of Γ_n in the **Precalc** algorithm with adding new elements to K. In this algorithm K is a list of integers and X is a set of irrationals such that $\|K[i] \cdot X[j]\| < \varepsilon$ for all i and for all $j < n$. The main part of the algorithm is the for loop (lines $4 - 9$). Each element of K is subtracted (added) from (to) every element of K and the new integer k_i that satisfies $\|k_i \cdot X[j]\| < \varepsilon$ for all $j < n$ are appended to K.

Precondition: K: set of integers,$n \in \mathbb{N}$, $\varepsilon > 0$, X: set of irrationals

```
1:  procedure REDUCE(K, n, ε, X)
2:      SORT(K)                          ▷ Sorting, every element ouccurs only once
3:      M ← dynamic array()
4:      for i = 1 → length(K) do
5:          for j = 1 → length(K) do
6:              APPEND(M, abs(K[i] − K[j]))      ▷ append abs(K[i] − K[j]) to M
7:              APPEND(M, abs(K[i] + K[j]))
8:          end for
9:      end for
10:     SORT(M)
11:     for i = 1 → length(M) do
12:         t ← TRUE
13:         for j = 1 → n do
14:             t ← t and (FRAC(M[i]·X[j]) < 2ε or FRAC(M[i]·X[j]) > 1−2ε)
15:         end for
16:         if t = FALSE then
17:             DELETE(M[i])                 ▷ Delete the i^th element of M
18:         end if
19:     end for
20:     APPEND(K, M)                         ▷ Append array M to K
21:     SORT(K)
22:     if K[1] = 0 then
23:         DELETE(K[1])                     ▷ Delete the zero value from K
24:     end if
25:     RETURN(K)
26: end procedure
```

Algorithm 5 Precalc

Description: The algorithm is based on **Lemma 8**. It generates Γ_n, a subset of Δ_Ω. In dimension n the set Γ_n contains exactly 2^n elements. Initially (line 2), the FindMMM algorithm is used. In higher dimensions $(2, 3, \ldots$ up to $m)$ the algorithm produces many integers from Δ_Ω by which Γ_n can be generated. M is a matrix with i rows. The i^{th} row contains the binary representation of i. (Note: the size of M is changing depending on the dimension.) To produce as many integers as possible the **Reduce** algorithm is used (see lines $10, 11$). If β goes to infinity then Δ_Ω should contain allmost all possible step-sizes, not just some. To solve Challenge 2, we set $\beta = 2^{12}$. With this choice of β the algorithm is able to generate the appropriate Γ_n up to 10 dimensions. For higher dimensions bigger β is needed.

```
 1: procedure PRECALC(m, ε, X, β)
 2:     T ← FINDMMM(X[1], ε)                                ▷ T is a dynamic array
 3:     for n = 2 → m do
 4:         T2 ← dynamic array()
 5:         N ← 0, T3 ← 0     ▷ N, T3 are arrays with 2^n elements, every element is 0
 6:         M ← 2^n × n matrix, the i^th row contains the binary representation of i
 7:         k ← 0, tmp ← 0, l ← 0, number ← 0
 8:         while TRUE do
 9:             if l = 2^n and number > β then
10:                 REDUCE(T2, n, ε, X)
11:                 REDUCE(T2, n, ε, X)
12:                 T ← T2
13:                 break                                  ▷ Leave the while loop
14:             end if
15:             for i = 1 → length(T) do
16:                 t ← TRUE
17:                 for j = 1 → n − 1 do
18:                     ok ← FRAC((k + T[i]) · X[j])
19:                     if ok > ε and ok < 1 − ε then
20:                         t ← FALSE
21:                         break                          ▷ Leave the for loop
22:                     end if
23:                     if t = TRUE then
24:                         k ← k + T[i]
25:                         break                          ▷ Leave the for loop
26:                     end if
27:                 end for
28:             end for
```

Algorithm 6 Precalc (contd.)

```
            number ← number + 1
            t ← TRUE
            for j = 1 → n do
                t ← t and (FRAC(k · X[j]) < ε or FRAC(k · X[j]) > 1 − ε)
            end for
            if t = FALSE then
                next
            end if
            t ← FALSE
            for i = 1 → length(T2) do
                if T2[i] = k − tmp then
                    t ← TRUE
                end if
            end for
            if t = FALSE then
                APPEND(T2, k − tmp)                ▷ append k − tmp to the array T2
            end if
            tmp ← k
            for i = 1 → 2^n do
                t ← TRUE
                for j = 1 → n do
                    if M[i, j] = 0 then
                        t ← t and (FRAC(k · X[j]) < ε)
                    else
                        t ← t and (FRAC(k · X[j]) > 1 − ε)
                    end if
                end for
                if t and N[i] = 0 then
                    N[i] ← 1
                    l ← l + 1
                    T3[l] ← k
                    if l = 2^n and n = m then
                        break(2)                   ▷ Leave while loop
                    end if
                end if
            end for
        end while
    end for
    RETURN(T3)
end procedure
```

4 Practical use of our methods

The real power of the presented methods is the ability to use them in a distributed way.

There are several fields of mathematics where the techniques shown in this paper can be applied. We used our methods in order to find high peak values of the Riemann-zeta function effectively. It is computationaly hard to find real t values where $|\zeta(1/2+it)|$ is high (see [11]). In 2004 Tadej Kotnik observed that large values of $|\zeta(1/2+it)|$ are expected when $t = \frac{2k\pi}{\log 2}$, where $k\frac{\log(p_i)}{\log(2)}$ are close to an integer for all primes $p_i > 2$ [6]. The methods shown in this paper can be used to find thousands of candidates within a few minutes where high values of $|\zeta(1/2+it)|$ are expected. We plan to continue our research in this direction.

5 Acknowledgement

The authors would like to thank Prof. Dr. Antal Járai for his very helpful comments, suggestions and to the anonymous reviewers for many constructive comments. The research of the first author was partially supported by the European Union and co-financed by the European Social Fund (ELTE Támop-4.2.2/B-10/1-2010-0030).

References

[1] F. Armknecht, C. Elsner, M. Schmidt, Using the Inhomogeneous Simultaneous Approximation Problem for Cryptographic Design. *AFRICACRYPT*, 2011, pp. 242–259. ⇒18

[2] A. Frank, É. Tardos, An application of simultaneous Diophantine approximation in combinatorial optimization, *Combinatorica*, **7,** 1 (1987) 49–66. ⇒18

[3] A. Y. Khinchin, *Continued Fractions*, Translated from the third (1961) Russian edition, Reprint of the 1964 translation, Dover, Mineola, NY, 1997. ⇒20

[4] Sh. Kim, S. Östlund, Simultaneous rational approximations in the study of dynamical systems, *Phys. Rev. A*, **34,** 4 (1986) 3426–3434. ⇒18

[5] C. Kimberling, Best lower and upper approximates to irrational numbers, *Elem. Math.*, **52,** 3 (1997) 122–126. ⇒20

[6] T. Kotnik, Computational Estimation of the order of $\zeta(1/2+it)$, *Math. Comp.*, **73,** 246 (2004) 949–956. ⇒33

[7] J. C. Lagarias, Best simultaneous Diophantine approximations I., Growth rates of best approximation denominators, *Trans. Am. Math. Soc.*, **272,** 2 (1982) 545–554. ⇒18

[8] J. C. Lagarias, Best simultaneous Diophantine approximations II., Behavior of consecutive best approximations, *Pacific J. Math.*, **102,** 1 (1982) 61–88. ⇒18
[9] J. C. Lagarias, The computational complexity of simultaneous Diophantine approximation problems, *SIAM J. Computing* **14,** 1 (1985) 196–209. ⇒18
[10] A. K. Lenstra, H. W. Lenstra Jr., L. Lovász, Factoring polynomials with rational coefficients, *Math. Ann.*, **261,** 4 (1982) 515–534. ⇒18, 21
[11] A. M. Odlyzko, The 10^{20}-th zero of the Riemann zeta function and 175 million of its neighbors, 1992 (unpublished) ⇒33
[12] V. T. Sós, G. Szekeres, Rational approximation vectors, *Acta Arithm.*, **49,** 3 (1988) 255–261. ⇒18

Received: April 10, 2013• Revised: June 8, 2013