On the diameter of tuples of weighted powers

C.E. van de Woestijne
Mathematical Institute, Rijksuniversiteit Leiden
P.O. Box 9512, 2300 RA Leiden, The Netherlands
e-mail: cvdwoest@wi.LeidenUniv.nl

July 27, 1998

Contents

Preface

My work on this Master's Thesis began in February 1997, when (then) faculty staff member Benne de Weger asked me to have a thorough look at a manuscript which he had been working on at times, but couldn't find time to finish. It was entitled *On the diameter of tuples of weighted powers*, and set out to explore two problems about *almost powers*, which we will introduce in Chapter 1.

For those wondering...

An almost power may be described as an integer number which is almost, but not quite, a perfect kth power for some k, i.e. of the form ax^k with a, x and k integers, and a such that $\sqrt[k]{a}$ is not an integer. In fact, we even require that a is kth power free, which means that it has not even a divisor with an integral kth root (of course, we exclude 1 as a divisor).

There are many questions that one can ask about (sets of) almost powers. The central consideration is that almost powers of the same kind (i.e. with the same exponent k) cannot be very close to one another. This intuition has given rise to some very impressive theorems — in my opinion —, known as the theorems of Dirichlet, Kronecker, and Thue-Siegel-Roth, which make quite general statements about how well roots of integers and rational numbers may be approximated by rationals. This area of research bears the name of Diophantine approximation, after Diophantus of Alexandria (although nowadays, one may ask just what it is that links this theory to Diophantus' work). This report contains applications of all these results, the last one in a generalized form proved by Schmidt.

In this work, we define two different settings in which these theorems may be applied and try to find out just how close certain concrete sets of almost powers can be. We first prescribe a small set of values for a, and then find out the minimal and maximal "space" needed to fit a set of almost powers with these a's as powerfree parts. Secondly, we stipulate that our numbers must be *consecutive*, and try to prove lower bounds on the values of the a's.

Apart from applying results from the (recent) past, my proofs, several of which are derived from Benne's paper, develop many ideas from scratch. This is referred to by mathematicians as 'giving a lot of information' or 'giving an elementary proof'. 'Elementary' in mathematics is often the opposite of 'simple' — it indicates that a proof requires no knowledge of other results to be understood, and that all the necessary work is done on the spot.

The remaining project history, and the acknowledgements

After Benne's request, the project lay in the fridge[1] till about September 1997, when I had finished most of my mathematical exams. By that time, Benne had switched to a job in another branch of industry, which greatly limited his possibilities for supervising my work. Professor R. Tijdeman then kindly took over the supervision task.

Benne's paper proved more than sufficient for half a year's work (for a poorly experienced student like me, that is). I found occasion to generalize some of his definitions, in particular the definition of diameter

[1] Dutch saying, meaning postponement

itself, which sometimes meant generalizing the proofs accordingly; some proofs needed a touch to be finished[2] or allowed simplification and clarification, those mathematicians' pets; and having proved some upper bounds naturally makes one ask for lower bounds, requiring completely different methods of proof. Finally, I took care of formalizing Benne's method of finding examples of large and small diameter, as well as some other algorithms which I had used in the process, and which I include in this report as Appendices.

I want to thank both Benne and Professor Tijdeman for the many fruitful discussions and criticisms, and for always having their doors open — which is a physical truth when referring to Professor Tijdeman's room at the Institute. I also appreciate having been allowed to progress at my own speed, and to explore various kinds of mathematical research activity — doing extensive computations, becoming acquainted to various Computer Algebra packages, penetrate into the number theoretical literature and applying the sometimes deep results found there, fiddling along with tiny index arrangement problems, to be *not* exhaustive. Together, this has resulted in a many-sided project.

I also want to thank Nils Bruin for his valuable assistance in mastering KASH and MapleV, and my roommates at 206a for providing a pleasant environment for both working and having 'coughee!'.

Finally (and foremost), I want to thank God, my Father in heaven, for having created the earth and all there is to it, for granting me life and faculties to investigate the mathematical character of His doings, among other activities (more or less so) — and enjoy it all.

Leiden, July 27, 1998
Christiaan van de Woestijne

All software developed in this report may be obtained in digital form from the author.

The KASH Computer Algebra package is freely available from the ftp site of the Technische Universität Berlin (host ftp.math.tu-berlin.de, directory /pub/algebra/Kant/Kash). More information may be obtained via WWW at http://www.math.tu-berlin.de/~kant/kash.html.

This report was typeset by AMS-LaTeX, using the report document class with the 10pt option, and the additional packages mapleenv, epsfig, calc, and url. The former package is used to process the output of the 'Export to LaTeX' feature in MapleV3.

[2]which is more than a finishing touch...

Chapter 1

Introduction and main results

1.1 The almost-power diameter of a positive integer

F. Göbel[1], observing that the three consecutive integers 48, 49 and 50 are respectively 3, 1 and 2 times a square, posed the question how exceptional this is. Namely, triples of 1, 2 and 3 times a square seem to have increasing differences. On the other hand, it is well known that for any squarefree $d \in \mathbb{Z}$, there are infinitely many pairs of integers (x, y) such that $x^2 - dy^2 = 1$ — the solutions of the ordinary Pell equation. In a more general setting, the problem may be formulated as follows.

Let $n \geq 3$ and $k \geq 2$ be given integers. Let $a_1, \ldots, a_n$ be given positive integers (the 'weights'). We assume that the numbers $a_1^{-1/k}, \ldots, a_n^{-1/k}$ are $\mathbb{Z}$-linearly independent. We define the *almost-power diameter* $\operatorname{diam}(X)$ of $X \in \mathbb{N}$ as the least possible diameter of a set $S \subset \mathbb{N}$ containing both X and, for $j = 1, \ldots, n$, a 'weighted power' $a_j x_j^k$. As may be expected, we define the diameter of a finite subset $S \subset \mathbb{R}$ as $\max_{s,t \in S} |s - t|$. It is clear that if any such set indeed has a minimal diameter, any of its weighted powers will be either $a_j x_j^{*k}$ or $a_j(x_j^* + 1)^k$, where $x_j^* = \lfloor \sqrt[k]{X/a_j} \rfloor$.

For example, if $k = 2$, $n = 3$, $a_1 = 3$, $a_2 = 2$, $a_3 = 1$, then $\operatorname{diam}(48) = \operatorname{diam}(49) = \operatorname{diam}(50) = 2$, corresponding to the above mentioned triple 48, 49, 50. From now on, we will abbreviate our term to just 'diameter'.

We want to know how small and how large $\operatorname{diam}(X)$ can be compared to X. Let us start with upper bounds. We introduce the following number, depending on n, k, and the weights a_i:

$$\vartheta = k \max_{I \subset \{1,\ldots,n\}} \frac{1}{|I| + 1} \sum_{j \in I} a_j^{1/k}. \tag{1.1}$$

We assume in the sequel that $a_1 > a_2 > \ldots > a_n$. Under this assumption we can write

$$\vartheta = k \max_{1 \leq i \leq n} \frac{1}{i + 1} \sum_{j=1}^{i} a_j^{1/k}.$$

For example, if $k = 2$, $n = 3$, $a_1 = 3$, $a_2 = 2$, $a_3 = 1$, then $\vartheta = \frac{2}{3}(\sqrt{3} + \sqrt{2}) = 2.0975\ldots$. The following theorem states that $\vartheta X^{1-\frac{1}{k}}$ is essentially the best uniform upper bound for $\operatorname{diam}(X)$.

Theorem 1 *For given $n, k, a_1, \ldots, a_n$ as defined above, we have*

$$\limsup_{X \to \infty} \frac{\operatorname{diam}(X)}{X^{1-1/k}} = \vartheta.$$

[1] A former colleague of Benne de Weger at the Universiteit Twente, Enschede, Holland

By making use of Schmidt's generalization of the Thue-Siegel-Roth theorem, it is possible to give adequate lower bounds for $\operatorname{diam}(X)$. In fact, we have

Theorem 2 *For every $\varepsilon > 0$ there exists a constant c, depending only on $\max_{1\le i\le n} a_i$ and ε, such that for all $X \in \mathbb{N}$ we have*

$$\operatorname{diam}(X) > cX^{1-\frac{1}{k}-\frac{1}{k(n-1)}-\varepsilon}.$$

As to the sharpness of this bound, we have

Theorem 3 *There exists a positive constant $\psi = \psi(n, k, a_1, \dots, a_n)$ such that, for every $\varepsilon > 0$, there are infinitely many $X \in \mathbb{N}$ with*

$$\operatorname{diam}(X) < (\psi + \varepsilon)X^{1-\frac{1}{k}-\frac{1}{k(n-1)}}.$$

If k is very large, then we can successfully apply the theory of linear forms in logarithms, and obtain an effective lower bound for $\operatorname{diam}(X)$. Indeed, we have the following result.

Theorem 4 *There exists a (very large) effectively computable constant C, depending only on $\max_{1\le i\le n} a_i$, such that for all $X \in \mathbb{N}$ we have*

$$\operatorname{diam}(X) > X^{1-C\log k/k}.$$

Clearly this starts to be anything at all only when $k \gg C\log C$.

Theorem 2 and the much weaker Theorem 4 are relatively straightforward applications of the above mentioned machinery. However, the proof of Theorem 1, besides using an extension of Kronecker's Theorem on simultaneous inhomogeneous Diophantine approximations, achieves a lot with elementary means and gives a lot of information, whereas theorem 3 is a simple application of Dirichlet's Theorem. This means that recent techniques of computational Diophantine approximation theory can be used for producing explicit examples of X for which the inequalities for $\operatorname{diam}(X)$ of Theorems 1 and 3 are true with a given accuracy.

It is possible in practice, although laborious, to solve effectively the problem of determining all $X \in \mathbb{N}$ with $\operatorname{diam}(X) \le \Delta_0$ for a given Δ_0 which is not too large. Namely, one considers for each $\Delta \in \mathbb{Z}$ with $|\Delta| \le \Delta_0$ the equations $a_j x_j^k - a_i x_i^k = \Delta$ for some fixed $i \in \{1, \dots, n\}$ and for all $j \in \{1, \dots, n\} \setminus \{i\}$. If $k \ge 3$ then each such equation is a Thue equation, that can be solved by standard, but laborious, methods (cf. [17]). If $k = 2$ we consider two such equations for different i's, and we have a system of simultaneous Pell equations, which can also be solved by standard methods (cf. [3]). However, in this work we do not explore this strategy. Instead, we present an algorithm based on lattice basis reduction to find concrete solutions to the Diophantine approximation systems used in the proofs of Theorems 1 and 3, and include samples of the output of this algorithm.

1.2 Consecutive almost powers

In this section, and the second part of this work, we take a somewhat different viewpoint. Now our object is to study finite sequences of consecutive positive integers, which are all divisible by sufficiently large perfect powers. Our interest is again initiated by the triple 48, 49, 50; these numbers are divisible by 16, 49, and 25, respectively, and thus their squarefree parts are relatively small.

Let $n \ge 2$ and $k \ge 2$ be given integers. For any $m \in \mathbb{N}$, we denote the kth power-free part of m by $\omega_k(m)$, and define the number $\lambda_{n,k}(m)$ as

$$\lambda_{n,k}(m) = \max_{i=0,1,\dots,n-1} \omega_k(m - i). \tag{1.2}$$

For example, $\lambda_{3,2}(50) = 3$.

One may notice immediately that on the one hand $\lambda_{n,k}(m) \le m$ for all $m \in \mathbb{N}$, and on the other hand there exist infinitely many $m \in \mathbb{N}$ with $\lambda_{n,k}(m) = m$ — just take any m which is squarefree. Thus there are no nontrivial upper bounds to $\lambda_{n,k}(m)$, and we will concentrate on lower bounds.

To investigate this problem, we try to find pairs (e,c), with $e \geq 0$ and $c > 0$, such that, for some particular n and k,

$$\liminf_{m\to\infty} \frac{\lambda_{n,k}(m)}{m^e} \leq c. \tag{1.3}$$

To show the existence of such numbers, we begin with remarking that (1.3) is trivially satisfied by the pair $(1,1)$ for all n and k, as $\omega_k(m) \leq m$ for all $m \in \mathbb{N}$. We may prove (1.3) for a smaller value of e by constructing a sequence of distinct positive integers $(m_i)_{i=1}^{\infty}$ such that $\lambda_{n,k}(m_i) \leq c m_i^{\beta}$ for all $i \geq 1$ and some constant $c > 0$.

Following these lines, we have proved the following results, the former two of which have been conjectured by B. de Weger.

Theorem 5

(i) *Let $n \geq 2$ and $k \geq 2$. Then (1.3) is satisfied by $e = 1 - \frac{k}{nk-1}$ and some effectively computable constant $c = c(n,k)$.*

(ii) *If n and k are odd, we may take $e = 1 - \frac{k}{nk-2}$.*

(iii) *If $n = k = 2$, we have (1.3) for $e = 0$ and $c = 2$.*

One may ask whether these upper bounds for e are sharp. We have some results regarding this question, which we list here. To begin with, we may apply the following theorem of J. Turk [16, Theorem 3], which was proved using the theory of linear forms in logarithms.

Theorem 6 (Turk) *Let $n \geq 2$ and $k \geq 3$. For all sufficiently large $m \in \mathbb{N}$, we have*

$$\lambda_{n,k}(m) > e^{(\log\log m)^{0.2}}.$$

For $n \geq 3$ and $k = 2$, we have

$$\lambda_{n,k}(m) > (\log m)^{0.04}.$$

Using estimates for the size of solutions of Thue equations, we have proved

Theorem 7 *Let $n \geq 2$ and $k \geq 3$. There exists a (large) effectively computable constant $c > 0$ such that*

$$\lambda_{n,k}(m) > c \cdot (\log m)^{\frac{1}{2k-1}}.$$

We have not been able to obtain an absolute lower bound on $\lambda_{n,k}(m)$ that is polynomial in m. However, the following result, almost exactly the desired one, follows if we assume that the *abc*-conjecture is true.

Theorem 8 *Let $n \geq 2$ and $k \geq 2$. If the abc-conjecture is true, (1.3) can be satisfied only if $e \geq 1 - \frac{k}{nk-n}$.*

Chapter 2

A tight upper bound for diam(X)

2.1 Notation and results

In the previous chapter, we announced an upper bound for the diameter of an integer, namely, our Theorem 1. We will now prove two propositions which together yield the theorem. We assume that we are given positive integers $n \geq 2$, $k \geq 2$, and $a_1 > a_2 > \ldots > a_n$ such that the numbers $a_i^{-1/k}$, $i = 1, \ldots, n$, are linearly independent over $\mathbb{Z}$.

Let X be a positive integer, and $x_j^* = \lfloor \sqrt[k]{X/a_j} \rfloor$, as designated in the Introduction. In the sequel, we use indices $i_1, \ldots, i_n$, resp. $j_1, \ldots, j_n$, to denote some ordering of the set $\{a_j x_j^{*k} \mid j = 1, \ldots, n\}$, resp. $\{a_j(x_j^* + 1)^k \mid j = 1, \ldots, n\}$. It is clear that $(i_1, i_2, \ldots, i_n)$ and $(j_1, j_2, \ldots, j_n)$ are permutations of $(1, 2, \ldots, n)$. We put $i_{n+1} = j_1$ and $j_0 = i_n$, and go on to consider subsets $\gamma_{\ell,r}$ of the union of these sets, defined by

$$\gamma_{\ell,r} = \left\{a_{i_\ell} x_{i_\ell}^{*k}, \ldots, a_{i_n} x_{i_n}^{*k}, a_{j_1}(x_{j_1}^* + 1)^k, \ldots, a_{j_r}(x_{j_r}^* + 1)^k\right\}$$

for $1 \leq \ell \leq n+1$, and $0 \leq r \leq n$. A subset $\gamma_{\ell,r}$ will be called *minimal* if $\{i_\ell, \ldots, i_n, j_1, \ldots, j_r\} \supset \{1, \ldots, n\}$, and this property is not shared by $\gamma_{\ell+1,r}$ and $\gamma_{\ell,r-1}$. We denote the ℓ and r indices of all minimal subsets, in increasing order, by $\ell_0 = 1, \ell_1, \ldots, \ell_m = n+1$ and $r_0 = 0, r_1, \ldots, r_m = n$ ($\gamma_{1,0}$ and $\gamma_{n+1,n}$ are clearly minimal in all cases) and thus find $1 \leq m \leq n$.

Lemma 2.1 *With the above notation, we have* $i_{\ell_s} = j_{r_{s+1}}$ *for* $s = 0, \ldots, m-1$.

Proof of Lemma 2.1. By definition, γ_{ℓ_s, r_s} is minimal, so that $\{i_{\ell_s+1}, i_{\ell_s+2}, \ldots, i_n, j_1, \ldots, j_{r_s}\}$ does not contain i_{ℓ_s}. By subsequently adjoining $j_{r_s+1}, j_{r_s+2}, \ldots$ to this set, we eventually obtain a set that contains i_{ℓ_s}, and thus contains a minimal subset, which must be $\gamma_{\ell_{s+1}, r_{s+1}}$ by definition. As $j_{r_{s+1}}$ was the last adjoined element, we conclude that $j_{r_{s+1}} = i_{\ell_s}$. □

We have obtained the following results. The first one implies the validity of the given upper bound, whereas the second one proves its tightness.

Proposition 2.2 *Let X be a positive integer, and x_j^* as defined above. Take the indices $i_1, \ldots, i_n$ and $j_1, \ldots, j_n$ such that*

$$a_{i_1} x_{i_1}^{*k} < \ldots < a_{i_n} x_{i_n}^{*k} \leq X < a_{j_1}(x_{j_1}^* + 1)^k < \ldots < a_{j_n}(x_{j_n}^* + 1)^k. \tag{2.1}$$

Then

$$\operatorname{diam}(X) \leq \tfrac{k}{m+1} \sum_{s=1}^{m} a_{j_{r_s}}^{1/k} X^{1-1/k} + O(X^{1-2/k}).$$

Proposition 2.3 *Let $T = \{t_1, \dots, t_v\} \subset \{1, \dots, n\}$ be such that $\frac{k}{v+1}\sum_{s=1}^{v} a_{t_s}^{1/k} < k a_{t_s}^{1/k}$ for $s = 1, \dots, v$. Then for all $\varepsilon > 0$, there are infinitely many positive integers which satisfy*

$$\operatorname{diam}(X) = \tfrac{k}{v+1}\sum_{s=1}^{v} a_{t_s}^{1/k} X^{1-1/k} + O(X^{1-1/k-(1-\varepsilon)/vk}); \tag{2.2a}$$

$$a_{t_1}x_{t_1}^{*k} < a_{t_2}x_{t_2}^{*k} < \dots < a_{t_v}x_{t_v}^{*k} \le X < a_{t_1}(x_{t_1}^{*}+1)^k < a_{t_2}(x_{t_2}^{*}+1)^k < \dots < a_{t_v}(x_{t_v}^{*}+1)^k, \tag{2.2b}$$

using the notations of Proposition 2.2.

From these propositions, we have

Corollary 2.4

(i) *For every $\varepsilon > 0$ there are only finitely many $X \in \mathbb{N}$ with* $\operatorname{diam}(X) > (\vartheta + \varepsilon)X^{1-1/k}$.

(ii) *For every $\varepsilon > 0$ there are infinitely many $X \in \mathbb{N}$ with* $\operatorname{diam}(X) > (\vartheta - \varepsilon)X^{1-1/k}$.

In the proof of this Corollary, we make use of the following Lemma, which provides an alternative definition for ϑ.

Lemma 2.5 *The maximality condition imposed on subsets I of $\{1, \dots, n\}$ in (1.1) is equivalent to the condition that*

$$k a_i^{1/k} \le \frac{k}{|I|+1}\sum_{j\in I} a_j^{1/k} \le k a_j^{1/k} \text{ for all } i \in \{1, \dots, n\} \setminus I \text{ and all } j \in I. \tag{2.3}$$

Proof of Lemma 2.5. Let $I \subset \{1, \dots, n\}$; write $m = |I|$. Let I' be obtained by leaving ℓ elements out of I, say $s_1, \dots, s_l$, and adding v elements, say $t_1, \dots, t_v$. Then

$$\begin{aligned}\sigma_I =_{def} \tfrac{k}{m+1}\textstyle\sum_{j\in I} a_j^{1/k} \ge \tfrac{k}{m+\ell-v+1}\sum_{j\in I'} a_j^{1/k} &\Leftrightarrow \left(1 - \tfrac{m+1}{m+\ell-v+1}\right)\sigma_I \ge \tfrac{k}{m+\ell-v+1}\left(\textstyle\sum_{j=1}^{\ell} a_{s_j}^{1/k} - \sum_{j=1}^{v} a_{t_j}^{1/k}\right)\\ &\Leftrightarrow (\ell - v)\sigma_I \ge k\left(\textstyle\sum_{j=1}^{\ell} a_{s_j}^{1/k} - \sum_{j=1}^{v} a_{t_j}^{1/k}\right).\end{aligned} \tag{2.4}$$

First assume that σ_I is maximal, i.e., greater than or equal to the corresponding sum for all $I' \subset \{1, \dots, n\}$. Then taking first $\ell = 0$ and $v = 1$ and then $\ell = 1$ and $v = 0$, we see that (2.3) is satisfied.

Next assume that we have (2.3); note that this implies the right hand side of (2.4). Then the above considerations show that σ_I is indeed maximal. □

Proof of Corollary 2.4. Let I be that subset of $\{1, \dots, n\}$ for which $\frac{k}{|I|+1}\sum_{j\in I} a_j^{1/k}$ is maximized, and thus equal to ϑ. The first claim follows from the former Proposition by considering the case that $\{j_{r_s} \mid s = 1, \dots, m\} = I$. As to the second one, our Lemma asserts that $T = I$ verifies the conditions of the latter Proposition, so that our claim follows from the definition of ϑ. □

Unfortunately, (2.2b) is not enough to prove that $T = \{j_{r_s} \mid s = 1, \dots, m\}$, which would constitute equivalence between maximality of $\operatorname{diam}(X)$ and the condition that $\frac{k}{m+1}\sum_{s=1}^{m} a_{j_{r_s}}^{1/k}$ be equal to ϑ. However, with the aid of Lemma 2.1, it is easy to verify that T does have this property if there is a u, $1 \le u \le n$, such that $i_1 = t_1, \dots, i_u = t_u, j_{n-u+1} = t_u, \dots, j_n = t_v$ — in other words, if the terms in (2.1) with indices from T are either less or greater than all other terms. Numerical experiments show that the majority of extremal diameter examples indeed have this property, whereas a minority has not. We refer to Section 4.2 for testable examples; issuing env.DoPrint := true; in KASH results in the Diam function displaying the order of the terms in (2.1) as well.

We have completed and simplified the proofs of the above Propositions from the manuscript by B. de Weger mentioned in the Preface. In particular, it still remained to prove that the proposed system of inhomogeneous Diophantine inequalities in proof of Proposition 2.3 has infinitely many solutions, where Kronecker's Theorem in principle only gives one.

2.2 Proof of validity

We make use of a less informative version of Proposition 2.2, which follows at once from the observation that x^k and $(x+1)^k$ differ by about x^{k-1}.

Lemma 2.6 *There exists a real number φ, depending on n, k, and the a_i, such that for all $X \in \mathbb{N}$*

$$\operatorname{diam}(X) < \varphi X^{1-1/k}.$$

Proof of Lemma 2.6. By the definition of x_j^* we have

$$\left|X - a_j x_j^k\right| \leq a_j\left((x_j^*+1)^k - x_j^{*k}\right) \leq a_j\left(k x_j^{*k-1} + \cdots + 1\right) \leq \varphi_1 x_j^{*k-1} \leq \varphi_2 X^{\frac{k-1}{k}}$$

for some constants φ_1, φ_2, and thus for any i and j we have

$$\left|a_i x_i^k - a_j x_j^k\right| \leq \left|a_i x_i^k - X\right| + \left|X - a_j x_j^k\right| \leq 2\varphi_2 X^{1-1/k},$$

which proves the lemma. $\square$

We introduce one more notation. For $j = 1, \ldots, n$, we define the numbers α_j by

$$X - a_j x_j^{*k} = \alpha_j X^{1-\frac{1}{k}}. \tag{2.5}$$

Note that $\alpha_j \geq 0$ and, by Lemma 2.6, $\alpha_j = O(1)$ as $X \to \infty$. Of course this says nothing about the order of magnitude of the α's for a particular X.

Proof of Proposition 2.2. Note that, for any $j \in \{1, \ldots, n\}$, $x_j^* \leq a_j^{-1/k} X^{1/k} < x_j^* + 1$, so that

$$x_j^* = \left(\frac{X}{a_j}\right)^{1/k} + O(1).$$

Together with (2.5), this implies

$$\begin{aligned}
a_j(x_j^*+1)^k - X &= a_j x_j^{*k} + k a_j x_j^{*k-1} + O(x_j^{*k-2}) - X \\
&= k a_j x_j^{*k-1} - \alpha_j X^{1-\frac{1}{k}} + O(x_j^{*k-2}) \\
&= \left(k a_j^{1/k} - \alpha_j\right) X^{1-\frac{1}{k}} + O(X^{1-\frac{2}{k}}).
\end{aligned} \tag{2.6}$$

Put

$$\begin{aligned}
\beta_0 &= X - a_{i_1} x_{i_1}^{*k}, \\
\beta_s &= a_{j_{r_s}}(x_{j_{r_s}}^* + 1)^k - a_{i_{l_s}} x_{i_{l_s}}^{*k} \qquad \text{for } s = 1, 2, \ldots, m-1, \\
\beta_m &= a_{j_n}(x_{j_n}^* + 1)^k - X.
\end{aligned}$$

Then by definition

$$\operatorname{diam}(X) = \min_{0 \leq s \leq m} \beta_s.$$

Using (2.5) and Lemma 2.1, we find

$$\begin{aligned}
\beta_0 &= \alpha_{i_{\ell_0}} X^{1-1/k} \\
&= \alpha_{j_{r_1}} X^{1-1/k}, \\
\beta_s &= \left(k a_{j_{r_s}}^{1/k} - \alpha_{j_{r_s}} + \alpha_{i_{\ell_s}}\right) X^{1-1/k} + O(X^{1-2/k}) \\
&= \left(k a_{j_{r_s}}^{1/k} - \alpha_{j_{r_s}} + \alpha_{j_{r_{s+1}}}\right) X^{1-1/k} + O(X^{1-2/k}) \qquad \text{for } s = 1, 2, \ldots, m-1, \\
\beta_m &= \left(k a_{j_{r_m}}^{1/k} - \alpha_{j_{r_m}}\right) X^{1-1/k} + O(X^{1-2/k}).
\end{aligned}$$

Now we have to find the α's such that $\min\limits_{0 \leq s \leq m} \beta_s$ is maximal. Notice that in the sum of the β_s the $\alpha's$ cancel out. As the minimum of the $\beta's$ cannot exceed the average, we have

$$\operatorname{diam}(X) \leq \tfrac{1}{m+1} \sum_{s=0}^{m} \beta_s = \tfrac{k}{m+1} \sum_{s=1}^{m} a_{j_{r_s}}^{1/k} X^{1-1/k} + O(X^{1-2/k}).$$

$\square$

2.3 Proof of tightness

We need a theorem of M. Sweet [15] about the number of solutions of certain systems of Diophantine inequalities, which we state as a lemma, in a somewhat simplified version which fits our purposes (e.g., we take $s = 1$ in Sweet's notation).

Lemma 2.7 *Let η_s and $\theta_s \in \mathbb{R}$ for $s = 1, \ldots, v$, $v \geq 1$. Suppose ψ is a decreasing function on the positive reals with values between 0 and 1, and assume that $\int_1^\infty \psi(t)^v dt$ diverges. Let $\lambda(k)$ denote the number of solutions $q \in \mathbb{Z}$, $\vec{p} = (p_1, \ldots, p_v) \in \mathbb{Z}^v$ of the system*

$$0 \leq q\,\eta_s - p_s - \theta_s < \psi(q), \qquad s = 1, \ldots, v,$$

with $1 \leq q \leq k$. Let g be a positive increasing (real) function such that

$$\left| \sum_{s=1}^{v} \eta_s p_s - q \right| \geq \frac{1}{P^v g(P)}$$

for all $q \in \mathbb{Z}$ and $\vec{p} \in \mathbb{Z}^v$, with $P = \max |p_i|$ sufficiently large (thus the matrix $[\eta_1, \ldots, \eta_v]^T$ has type $\leq g$ in Sweet's terminology). If

$$F(t)^{v+2} := \psi(t)^v\ t\ g(t^{1/v})^{-1}$$

is increasing to infinity, we have

$$\lambda(k) = 2 \int_1^k \psi(t)^v dt + O\left(\int_1^k \psi(t)^v F(t)^{-1} dt \right).$$

Proof of Lemma 2.7. See [15]. □

Proof of Proposition 2.3. As may be seen from the proof of Proposition 2.2, $\operatorname{diam}(X)$ is maximized if all β's are equal. Therefore we study the linear system

$$\gamma_1 = k a_{t_s}^{1/k} - \gamma_s + \gamma_{s+1} = k a_{t_v}^{1/k} - \gamma_v \qquad \text{for } s = 1, 2, \ldots, v-1, \tag{2.7}$$

which is obtained from the system $\beta_0 = \beta_s$ by leaving the lower order terms out. This linear system has v equations in the v unknowns γ_s ($s = 1, 2, \ldots, v$). Clearly the determinant of the system is nonzero, so there is exactly one solution. Furthermore, this solution is not hard to find, as our linear system implies that $\gamma_1 = \frac{k}{v+1} \sum_{s=1}^{v} a_{t_s}^{1/k}$.

Now our aim is to find $x_{t_s}^* \in \mathbb{Z}$ such that the corresponding α_{t_s} are good approximations of the γ_s. Let b be such that $\{b^{-1/k}\} \cup \{a_{t_s}^{-1/k} \mid s = 1, \ldots, v\}$ is a $\mathbb{Z}$-linear independent set; in most cases, $b = 1$ will suffice, or $b = \frac{2}{3}$ if $a_{t_s} = 1$ already for some s. By the theorem of Kronecker on inhomogeneous simultaneous Diophantine approximation, there are positive integers y and $x_{t_s}^*$ (for $s = 1, 2, \ldots, v$) such that, for all positive ε, we have

$$\left| \left(\frac{b}{a_{t_s}} \right)^{1/k} y - x_{t_s}^* - \frac{\gamma_s}{k a_{t_s}^{1/k}} \right| < y^{\frac{-1+\varepsilon}{v}}. \tag{2.8}$$

In fact, we can apply Lemma 2.7 to show that there are *infinitely many* solutions to this system.

Set $\psi(t) = t^{\frac{-1+\varepsilon}{v}}$; it is clear that we have $\lim_{T \to \infty} \int_1^T t^{-1+\varepsilon} dt = \infty$. Take further $\eta_s = (\frac{b}{a_{t_s}})^{1/k}$, and $\theta_s = \frac{\gamma_s}{k a_{t_s}^{1/k}}$ ($s = 1, \ldots, v$). In order to verify the assertion about g, we apply a generalization of Roth's Theorem by Schmidt [14, Corollary VI.1E] to find that for any $\delta > 0$ there are only finitely many $q \in \mathbb{Z}$ and $\vec{p} = (p_1, p_2, \ldots, p_v) \in \mathbb{Z}^v$ such that

$$\left| \sum_{s=1}^{v} \eta_s p_s - q \right| < P^{-v-\delta},$$

where $P = \max |p_i| > 0$. Choose any δ with $0 < \delta < v\varepsilon$ and take $g(t) = t^\delta$. Then

$$F(t)^{v+2} = \psi(t)^v\ t\ g(t^{1/v})^{-1} = t^{\varepsilon - \frac{\delta}{v}},$$

increases to infinity with t. As all conditions are verified, Lemma 2.7 allows us to conclude that the number of solutions $\lambda(\kappa)$ of (2.8) with $y < \kappa$ satisfies

$$\lambda(\kappa) = 2\int_1^{\kappa} t^{-1+\varepsilon} dt + O\left(\int_1^{\kappa} t^{-1+\varepsilon} F(t)^{-1} dt\right), \tag{2.9}$$

which clearly grows to infinity as κ does.

We have thus found an infinitude of solutions $(y, x_{t_1}^*, \dots, x_{t_v}^*) \in \mathbb{Z}^{v+1}$ to (2.8). For any such solution, put $X = by^k$. Now we have

$$\begin{aligned} \alpha_{t_s} X^{1-1/k} &= X - a_{t_s} x_{t_s}^{*k} \\ &= by^k - a_{t_s}\left(\left(\frac{b}{a_{t_s}}\right)^{1/k} y - \frac{\gamma_s}{k a_{t_s}^{1/k}} + O\left(y^{\frac{-1+\varepsilon}{v}}\right)\right)^k \\ &= \gamma_s b^{1-1/k} y^{k-1} + O(y^{k-1-(1-\varepsilon)/v}) \\ &= \gamma_s X^{1-1/k} + O(X^{1-1/k-(1-\varepsilon)/vk}), \end{aligned} \tag{2.10}$$

so that $\alpha_{t_s} = \gamma_s + O(X^{(-1+\varepsilon)/vk})$, as required.

We claim that there exists a constant $c > 0$ such that $\gamma_v \geq c$, and $\gamma_s - \gamma_{s+1} \geq c$ for $s = 1, 2, \dots, v-1$. To see this, note first that $\gamma_s = \gamma_{s+1}$ for an s, resp. $\gamma_v = 0$, is impossible because of the $\mathbb{Z}$-linear independence of the $a_i^{-1/k}$. Furthermore, if one such number is negative, we have a $t_0 \in T$ with

$$k a_{t_0}^{1/k} < \gamma_1$$

by (2.7), contradicting the assumptions of the Proposition. Our claim follows.

Substituting our c in (2.7), we immediately have the existence of a constant $c' > 0$ with

$$\alpha_{t_s} - \alpha_{t_{s+1}} > c', \quad \alpha_{t_v} > c'.$$

This implies

$$a_{t_1} x_{t_1}^{*k} < a_{t_2} x_{t_2}^{*k} < \dots < a_{t_v} x_{t_v}^{*k} < X.$$

Using (2.6), we also find

$$X < a_{t_1}(x_{t_1}^* + 1)^k < a_{t_2}(x_{t_2}^* + 1)^k < \dots < a_{t_v}(x_{t_v}^* + 1)^k.$$

It is now easy to prove the correct lower bound for the diameter: by (2.7) and (2.10), we have

$$\begin{aligned} \beta_s &= \left(k a_{t_s}^{1/k} - \alpha_{t_s} + \alpha_{t_{s+1}}\right) X^{1-1/k} + O(X^{1-2/k}) \\ &= \left(k a_{t_s}^{1/k} - \gamma_s + \gamma_{s+1}\right) X^{1-1/k} + O(X^{1-1/k-(1-\varepsilon)/vk}) \\ &= \gamma_1 X^{1-1/k} + O(X^{1-1/k-(1-\varepsilon)/vk}), \end{aligned}$$

for $s = 1, \dots, v-1$, whereas the same estimate for β_0 and β_v may be derived analogously. Hence we find

$$\operatorname{diam}(X) = \min_{0 \leq s \leq v} \beta_s = \tfrac{k}{v+1} \sum_{s=1}^{v} a_{t_s}^{1/k} X^{1-1/k} + O(X^{1-1/k-(1-\varepsilon)/vk}).$$

$\square$

Chapter 3

Lower Bounds for diam(X)

3.1 The Bound

Whereas the proof of Theorem 1 achieves much by elementary means, we will need very powerful results in our proof of a lower bound to diam(X) of the same form, viz. polynomial in X. Fortunately, we can show that this bound is sharp in some sense. The proofs in this section have been generalized from proofs of B. de Weger.

Let us indeed assume that $a_1 > a_2 > \ldots > a_n$, and put

$$\psi = k a_n^{1/k(n-1)} \left(a_1^{1/k} + a_2^{1/k}\right).$$

Then, by virtue of the well-known Dirichlet theorem on simultaneous approximation, we have

Theorem 3.1 *For every $\varepsilon > 0$ there are infinitely many $X \in \mathbb{N}$ with*

$$\operatorname{diam}(X) < (\psi + \varepsilon) X^{1-\frac{1}{k}-\frac{1}{k(n-1)}}.$$

For our standard example $k = 2$, $n = 3$, $a_1 = 3$, $a_2 = 2$, $a_3 = 1$, we find $\psi = 2\sqrt[4]{1}(\sqrt{3} + \sqrt{2}) = 6.293\ldots$.

Proof of Theorem 3.1. The theorem of Dirichlet on homogeneous simultaneous Diophantine approximation [14, Corollary II.1C] guarantees the existence of infinitely many solutions $x_1, x_2, \ldots, x_n$ in $\mathbb{Z}$ of the inequalities

$$\left| x_i - \left(\frac{a_n}{a_i}\right)^{1/k} x_n \right| < \frac{1}{x_n^{1/(n-1)}} \qquad (i = 1, \ldots, n-1). \tag{3.1}$$

Now

$$\begin{aligned} |a_n x_n^k - a_j x_j^k| &= |a_n^{1/k} x_n - a_j^{1/k} x_j| |a_n^{1-1/k} x_n^{k-1} + a_n^{1-2/k} a_j^{1/k} x_n^{k-2} x_j + \cdots + a_j^{1-1/k} x_j^{k-1}| \\ &< a_j^{1/k} x_n^{-1/(n-1)} \left(k a_n^{1-1/k} x_n^{k-1} + O(x_n^{k-2})\right), \end{aligned}$$

and hence

$$\begin{aligned} |a_i x_i^k - a_j x_j^k| &< k a_n^{\frac{1}{k(n-1)}} (a_i^{1/k} + a_j^{1/k}) (a_n x_n^k)^{1-\frac{1}{k}-\frac{1}{k(n-1)}} + O(x_n^{k-2-\frac{1}{n-1}}) \\ &< (\psi + \varepsilon) X^{1-\frac{1}{k}-\frac{1}{k(n-1)}}, \end{aligned}$$

upon choosing $X = a_n x_n^k$, as we have done before. $\square$

The following theorem states that we cannot expect to improve our exponent $1 - \frac{1}{k} - \frac{1}{k(n-1)}$.

Theorem 3.2 *For every $\varepsilon > 0$ there exists a constant c, depending only on $\max_{1 \le i \le n} a_i$ and ε, such that for all $X \in \mathbb{N}$ we have*

$$\operatorname{diam}(X) > c X^{1-\frac{1}{k}-\frac{1}{k(n-1)}-\varepsilon}.$$

J. Turk has proved essentially the same result [16, Theorem 1]. However, as he uses a different definition of diameter, the details of his proof differ from ours.

Proof of Theorem 3.2. Let $\varepsilon > 0$, and consider $X \in \mathbb{N}$ such that $\mathrm{diam}(X) < X^{1-1/k-\frac{1}{k(n-1)}-\varepsilon}$.

First let us restrict ourselves to X of the form $a_{i_0}x^k$ for some $x \in \mathbb{N}$ and some $i_0 \in \{1, \dots, n\}$. For each $i \in \{1, \dots, n\} \setminus \{i_0\}$ let x_i be the integer such that $|a_{i_0}x^k - a_i x_i^k|$ is minimal. Then clearly there are constants c, c' such that for all i we have

$$|x_i - \left(\frac{a_{i_0}}{a_i}\right)^{1/k} x| < c\,|a_{i_0}x^k - a_i x_i^k|\,x^{-(k-1)} < c\ \mathrm{diam}(X)x^{-(k-1)} < c'\,x^{-\frac{1}{n-1}-\varepsilon} < x^{-\frac{1}{n-1}-\frac{1}{2}\varepsilon}$$

if x is large enough. Now Schmidt's generalization of the Thue-Siegel-Roth Theorem [14, Corollary VI.1C] tells us that there are only finitely many such x. The result follows.

Next consider general X. Define the integers x_i^* $(i = 1, \dots, n)$ and the indices $i_1, \dots, i_n$ and $j_1, \dots, j_n$ as in Proposition 2.2. Write $\varphi_\ell = a_{i_\ell} x_{i_\ell}^{*k}$ and $\psi_r = a_{j_r}(x_{j_r}^* + 1)^k$ for $\ell, r = 1, \dots, n$. There are three cases:

(i) If $\mathrm{diam}(X) = \psi_n - X$, then $\mathrm{diam}(X) \geq \mathrm{diam}(\psi_n - 1) > \psi_n^{1-1/k-1/k(n-1)-\varepsilon} > X^{1-1/k-1/k(n-1)-\varepsilon}$.

(ii) If $\mathrm{diam}(X) = \psi_r - \varphi_\ell$ for some ℓ and r, then $\mathrm{diam}(X) = \mathrm{diam}(\varphi_n)$, i.e. of the form discussed above.

(iii) If $\mathrm{diam}(X) = X - \varphi_1$, then note that we have already shown that $|\varphi_n - \varphi_1| = \mathrm{diam}(\varphi_n) > \varphi_n^{1-1/k-1/k(n-1)-\varepsilon}$, so that $\mathrm{diam}(X) = (X - \varphi_n) + (\varphi_n - \varphi_1) > X - \varphi_n + \varphi_n^{1-1/k-1/k(n-1)-\varepsilon} > X^{1-1/k-1/k(n-1)-\varepsilon}$. Here we use the inequality $B - A > B^\alpha - A^\alpha$, equivalent to $B - B^\alpha > A - A^\alpha$, which holds whenever $B > A > 1$ and $0 < \alpha < 1$.

Hence we see that X of general form with very small diameter can only occur in the (finite) neighbourhood of special X with even smaller diameter. This proves the Theorem. □

3.2 Ways to obtain effective lower bounds

3.2.1 The problem

We have seen that Schmidt's theorem can be applied in order to get a lower bound for $\mathrm{diam}(X)$: we have, for any positive ε,

$$\mathrm{diam}(X) > cX^{1-\frac{1}{k}-\frac{1}{k(n-1)}-\varepsilon}$$

for some constant c depending only on $\max_{1\leq i\leq n} a_i$ and ε. Unfortunately, due to the ineffective nature of Schmidt's work, there is no known algorithm to calculate c explicitly for a given ε, and no ground to make conjectures about the existence — let alone the value — of

$$\liminf_{X\to\infty} \frac{\mathrm{diam}(X)}{X^{1-\frac{1}{k}-\frac{1}{k(n-1)}}}.$$

One might want to know whether it is possible to give *explicit* lower bounds.

3.2.2 Linear forms in logarithms

There are two approaches that one may take. The first one is to apply the theory of linear forms in logarithms of algebraic numbers, as developed by Baker [2]. This results in the following theorem.

Theorem 3.3 *Let $k \geq 3$. There exists a (very large) effectively computable constant C, depending only on a_{n-1} and a_n, such that for all $X \in \mathbb{N}$ we have*

$$\mathrm{diam}(X) > X^{1-C\log k/k}.$$

Clearly this starts to be anything at all only when $k \gg C \log C$.

Proof of Theorem 3.3. This is an easy consequence of equation (1) of Proposition 1 in [16]: just take $m = k$ and $n = 2$, and take the interval $[N, N+K]$ equal to the smallest set around X containing all required almost powers, so that $K = \mathrm{diam}(X)$ (recall our definition of diameter). The result follows immediately. $\square$

It is well possible that recent developments have improved on the value of the constant C.

3.2.3 Effective irrationality measures

The second approach makes use of results concerning (possibly simultaneous) effective irrationality measures for certain sets of algebraic numbers. The most recent results along these lines that we have found, have been obtained by Bennett [5]; for other authors, we refer to his bibliography. For a survey of the methods used in the proof of such results, see [8]. In the cited paper, Bennett shows, for instance, that

$$\max\left\{\left|\sqrt{2} - p_1/q\right|, \left|\sqrt{3} - p_2/q\right|\right\} > q^{-1.79155} \tag{3.2}$$

for all p_1, p_2, and $q \in \mathbb{Z}$ with $q > q_0$. The number $\lambda = 1.79155$ is a simultaneous irrationality measure for $\{\sqrt{2}, \sqrt{3}\}$.

The sets of numbers discussed by Chudnovsky and subsequent authors are sets of kth roots of rationals which are all reasonably close to 1 with respect to a common denominator[1] Equation (3.2) was obtained by executing a suitable linear transformation on such a set, of cardinality 2, thus reducing the common denominator to 1.

When comparing (3.2) to the Dirichlet approximation in the proof of Theorem 3.1, viz. (3.1), it is clear that we cannot apply this result directly to our problem, except when $n = 2$: instead of common denominators, the numbers whose kth roots appear in the approximation system should have common numerators. However, consider finding X of small diameter in the following setting: $n = 3$, $k = 2$, $a_1 = 6$, $a_2 = 3$, $a_3 = 2$. If we choose x_1 instead of x_3 as the 'anchor variable' in (3.1), it follows that we try to find $x_i \in \mathbb{Z}$ such that

$$\left|x_2 - \sqrt{2}x_1\right| \text{ and } \left|x_3 - \sqrt{3}x_1\right|$$

are simultaneously small. Now Bennett's result shows that at least one of these expressions must be $> x_1^{-0.79155}$ if x_1 is sufficiently large. Analogously to the proof of Theorem 3.2, we now find that

$$\mathrm{diam}(X) > X^{1-\frac{1}{k}-\frac{0.79155}{k}} = X^{0.10423}$$

if X is sufficiently large (the constant -0.79155 plays the role of $-\frac{1}{n-1} - \frac{1}{2}\varepsilon$ in the proof).

More generally, we have

Lemma 3.4 *Suppose n, k and a_i are given as above, and suppose that we have a simultaneous irrationality measure of λ for the following set of numbers: $\{(a_i/a_n)^{1/k} \mid i = 1, \ldots, n-1\}$. Then we have*

$$\mathrm{diam}(X) > X^{1-\frac{1}{k}+\frac{-\lambda+1}{k}} = X^{1-\frac{\lambda}{k}},$$

if X is sufficiently large. Furthermore, this lower bound for the exponent of X is trivial, resp. optimal precisely when λ is.

Proof of Lemma 3.4. Our example shows how to derive the first assertion. As concerns the second claim, we have $1 = X^0 \leq \mathrm{diam}(X)$ as a trivial lower bound; this corresponds to $\lambda = k$, which is the trivial upper bound for λ. On the other hand, the optimal exponent for X is $1 - 1/k - 1/k(n-1) - \delta$. This corresponds to $\lambda = 1 + 1/(n-1) + k\delta$, which is optimal for λ as well (cf. [5]). $\square$

For more examples, including higher values of both n and k, we refer to [5].

---

[1] That is to say, they are of the form $1 + \frac{a_i}{N}$, with the a_i integers with $0 < |a_i| < |N|$.

Chapter 4

How to find examples of large or small diameter

4.1 Preliminaries and picture

Before entering into our chase of examples, we will make some remarks that follow from our definition of the diameter of a positive integer (page 5). The diameter of $X \in \mathbb{Z}$ is calculated by constructing a set around X consisting of almost-powers satisfying certain requirements — viz. we dictate their kth power-free parts — and then finding the smallest such set.

This construction implies that probably the diameter function will be constant on rather large intervals. In particular, many extremal values of the diameter function will be taken on on intervals (of integers) rather than points.

Our algorithm for finding X with small diameter produces values of X of the form $a_{i_0}x^k$ for some $i_0 \in \{1, \ldots, n\}$ and some $x \in \mathbb{N}$. It is easily seen that X's that are not of this form cannot yield smaller diameters — however, their diameters may equal those of X that are. The reader may check this for himself, using the values of the x_j^* which are also listed in our tables (cf. the proofs of Proposition 2.2 and Theorem 3.2).

However, from the proof of Theorem 3.2 it follows that there are values of X of general form whose diameter is larger than that of all neighbouring X of almost-power form. Therefore we will have to augment our algorithm in this case to search the neighbourhood of the points found, in order to find really the largest examples.

To illustrate our remarks, in Figure 4.1 we show a plot of the diameter function for one of the settings that we will explore below: $n = 4$, $k = 3$, $a_1 = 7$, $a_2 = 5$, $a_3 = 3$, $a_4 = 2$.

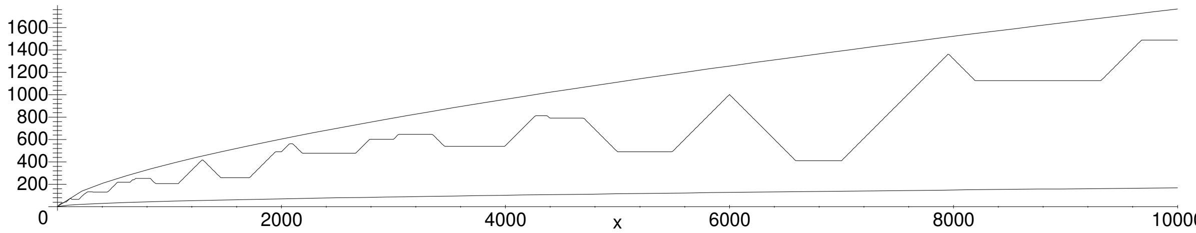

Figure 4.1: The diameter function for $n = 4$, $k = 3$, $a_1 = 7$, $a_2 = 5$, $a_3 = 3$, $a_4 = 2$, enclosed between $\vartheta X^{1-1/k} = 3.788\ldots X^{2/3}$ above and $X^{1-1/k-1/k(n-1)} = X^{5/9}$ below.

4.2 At work

In the sequel, we generate solutions to the Diophantine approximation systems on which the proofs of Theorems 1 and 3 are based, by finding short elements in suitable lattices (cf. [18] for an overview of this technique). The computational burden of such searches may be reduced by using suitable bases for the lattices used, e.g. a L^3-reduced basis. This has the additional notational benefit that the coefficients of the interesting lattice points with respect to the chosen basis will be small. However, we leave the actual implementation of the search to the number theory package KASH, which has a built-in feature, termed lattice enumeration, for finding all points $\boldsymbol{x}$ of a lattice such that

$$0 \le L \le \|\boldsymbol{x} - \boldsymbol{x_0}\| \le R,$$

for a lower bound L, an upper bound R, and a reference vector $\mathbf{x_0}$ (which is not necessarily a lattice point). For the KASH-source code used, we refer to Appendix A.1.

We take again our standard example, $k = 2$, $n = 3$, $a_1 = 3$, $a_2 = 2$, $a_3 = 1$, and inform KASH of our intentions by typing env := DiamEnvironment (3, 2, [3,2,1]). Then, as noted above,

$$\vartheta = \frac{2}{3}\sqrt{3}(1+\sqrt{2}) \approx 2.0975096, \quad \psi = 2\sqrt[4]{1}(\sqrt{3}+\sqrt{2}) \approx 6.2925287.$$

Let us first look for X in the order of magnitude of 10^{12} with small $\mathrm{diam}(X)$. Proceeding as in the proof of Theorem 3.1, we look for $x_1, x_2, x_3 \in \mathbb{Z}$ which have the magnitude of $X^{1/k}$, i.e. 10^6, and with

$$\begin{cases} |x_3 - x_1/\sqrt{3}| & \approx 10^{-3} \\ |x_3 - x_2/\sqrt{2}| & \approx 10^{-3} \end{cases}.$$

By Theorem 3.2, we can assume that this system has almost no solutions if we replace 10^{-3} by a smaller value, unless we allow the x_i to grow beyond 10^6.

We consider the lattice

$$\Gamma = \{\mathcal{A}\boldsymbol{x} | \boldsymbol{x} \in \mathbb{Z}^3\},$$

where the matrix $\mathcal{A}$ is given by

$$\mathcal{A} = \begin{bmatrix} 10^9 & 0 & -[10^9/\sqrt{3}] \\ 0 & 10^9 & -[10^9/\sqrt{2}] \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 10^9 & 0 & -577350269 \\ 0 & 10^9 & -707106781 \\ 0 & 0 & 1 \end{bmatrix}.$$

Here square brackets mean rounding to the nearest integer. For any $x_1, x_2, x_3 \in \mathbb{Z}$ of magnitude *at most* 10^6, we expect at least one component of the lattice point

$$\mathcal{A}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \approx \begin{pmatrix} 10^9(x_1 - x_3/\sqrt{3}) \\ 10^9(x_2 - x_3/\sqrt{2}) \\ x_3 \end{pmatrix}$$

to be *not much less* than 10^6 in magnitude, by the discussion above. As the length of a vector grows almost linearly with its largest component, we can assume that all interesting triples correspond to lattice elements that are among the shortest in the lattice.

An L^3-reduced basis of our lattice can be found by the L^3-algorithm, and in fact is given by the vectors

$$\boldsymbol{b_1} = \begin{pmatrix} -509249 \\ -608801 \\ 134421 \end{pmatrix}, \quad \boldsymbol{b_2} = \begin{pmatrix} 28237 \\ 106413 \\ -979473 \end{pmatrix}, \quad \boldsymbol{b_3} = \begin{pmatrix} 891562 \\ -960662 \\ -150898 \end{pmatrix}.$$

Note that indeed they have the magnitude of 10^6. Now $-\boldsymbol{b_2}$, for example, comes from x_1, x_2, x_3 satisfying $\boldsymbol{b_2} = \mathcal{A}(x_1, x_2, x_3)^t$, from which we compute $x_1 = 565499, x_2 = 692592, x_3 = 979473$; we may thus take $X = 979473^2 = 959367357729$. Let us calculate its diameter: we have

$$\begin{aligned} 3x_1^2 &= 959367357003 < X; \\ 2x_2^2 &= 959367356928 < 3x_1^2; \\ \text{and } x_3^2 &= 959367357729 = X. \end{aligned}$$

Thus we have $\mathrm{diam}(X) \leq X - 2x_2^2 = 801$; one easily verifies that the other possibilities do not yield a smaller value for the diameter. Now 801 is small, since $\mathrm{diam}(X)/\sqrt[4]{X} = 0.8093\ldots$ is considerably less than ψ. Taking other linear combinations of the $\boldsymbol{b_i}$ we find lattice points leading to other interesting examples. In Table 4.1, we have listed all examples X which follow from points in the above lattice — by taking $X = x_3^2$ — and satisfy $3 \cdot 10^{11} < X < 3 \cdot 10^{12}$.

lattice point	x_1	x_2	x_3	$\mathrm{diam}(X)$	$\mathrm{diam}(X)/\sqrt[4]{X}$
$-3\boldsymbol{b_1} - \boldsymbol{b_2} - \boldsymbol{b_3}$	419796	514143	727108	5234	$6.138\ldots$
$-2\boldsymbol{b_1} - \boldsymbol{b_2} - \boldsymbol{b_3}$	497404	609193	861529	4850	$5.225\ldots$
$-\boldsymbol{b_1} - \boldsymbol{b_2} - \boldsymbol{b_3}$	575012	704243	995950	5666	$5.678\ldots$
$-\boldsymbol{b_2} - \boldsymbol{b_3}$	652620	799293	1130371	6498	$6.112\ldots$
$-\boldsymbol{b_1} + \boldsymbol{b_2} - \boldsymbol{b_3}$	555986	680941	962996	5626	$5.733\ldots$
$\boldsymbol{b_2} - \boldsymbol{b_3}$	478378	585891	828575	4890	$5.372\ldots$
$\boldsymbol{b_1} + \boldsymbol{b_2} - \boldsymbol{b_3}$	400770	490841	694154	4138	$4.967\ldots$
$2\boldsymbol{b_1} + \boldsymbol{b_2} - \boldsymbol{b_3}$	323162	395791	559733	3443	$4.602\ldots$
$-3\boldsymbol{b_1} - 2\boldsymbol{b_2}$	898174	1100034	1555683	6339	$5.082\ldots$
$-2\boldsymbol{b_1} - 2\boldsymbol{b_2}$	975782	1195084	1690104	3756	$2.889\ldots$
$-3\boldsymbol{b_1} - \boldsymbol{b_2}$	332675	407442	576210	2775	$3.656\ldots$
$-2\boldsymbol{b_1} - \boldsymbol{b_2}$	410283	502492	710631	2106	$2.498\ldots$
$-\boldsymbol{b_1} - \boldsymbol{b_2}$	487891	597542	845052	939	$1.021\ldots$
$-\boldsymbol{b_2}$	565499	692592	979473	801	$0.809\ldots$
$\boldsymbol{b_1} - \boldsymbol{b_2}$	643107	787642	1113894	2908	$2.755\ldots$
$2\boldsymbol{b_1} - \boldsymbol{b_2}$	720715	882692	1248315	5550	$4.967\ldots$

Table 4.1: Examples of magnitude 10^{12} with small diameter (output of KASH-command DiamSmall (env, DiamPsi (env), 3*10^11, 3*10^12, 10^9))

Next we look for X in the order of magnitude of 10^{12} with large $\mathrm{diam}(X)$; we follow the proofs of Corollary 2.4 and Proposition 2.3. It is easy to verify that the index subset I such that $\frac{k}{|I|+1}\sum_{j\in I} a_j^{1/k}$ is maximal is equal to $\{1,2\}$. Thus we have $T = \{1,2\}$ and $m = 2$; however, we may take either $t_1 = 1$ and $t_2 = 2$ or $t_1 = 2$ and $t_2 = 1$. We choose the former possibility; it may be noted that reversing the order changes the values of the γ_s that we are about to calculate, and thus yields other examples, as these are constructed from solutions of (2.8).

Solving the resulting linear system (2.7), we find $\gamma_1 = \frac{2}{3}(\sqrt{3}+\sqrt{2})$, and $\gamma_2 = \frac{2}{3}(2\sqrt{2}-\sqrt{3})$. As $a_1 \neq 1 \neq a_2$, we may take $b = 1$. Hence we want $x_1, x_2, y \in \mathbb{Z}$ with

$$\begin{cases} |x_1 - y/\sqrt{3} + \frac{\sqrt{3}+\sqrt{2}}{3\sqrt{3}}| & \approx 10^{-3} \\ |x_2 - y/\sqrt{2} + \frac{2\sqrt{2}-\sqrt{3}}{3\sqrt{2}}| & \approx 10^{-3}, \end{cases}$$

as $y^{\frac{-1}{2}} \approx (10^6)^{-1/2}$. Thus it turns out that we can use the same lattice as above, but now look for lattice points in the neighborhood of the (non-lattice) point

$$\boldsymbol{y} = \begin{pmatrix} \left[10^9 \frac{\sqrt{3}+\sqrt{2}}{3\sqrt{3}}\right] \\ \left[10^9 \frac{2\sqrt{2}-\sqrt{3}}{3\sqrt{2}}\right] \\ 0 \end{pmatrix} = \begin{pmatrix} 605498860 \\ 258418376 \\ 0 \end{pmatrix}.$$

According to KASH, the lattice point nearest to $\boldsymbol{y}$ is

$$\boldsymbol{b_0} = \begin{pmatrix} 605051719 \\ 258857831 \\ 147949 \end{pmatrix} = \mathcal{A} \begin{pmatrix} 85419 \\ 104616 \\ 147949 \end{pmatrix}.$$

Thus $x_1 = 85419, x_2 = 104616, y = 147949$ is of interest: taking $X = 147949^2$, we find $\mathrm{diam}(X) = 310082$, which is large, since $310082/147949 = 2.0958\ldots$ is almost ϑ. Looking at other lattice points in the neighborhood produces the interesting examples listed in Table 4.2, all of which satisfy $\mathrm{diam}(X)/X^{1/2} > \vartheta - 0.003$. As

lattice point	x_1	x_2	y	X	diam(X)	diam(X)/$\sqrt{X}$
$\boldsymbol{b_0-b_1-b_2}$	573309	702157	993001	986050986799	2081501	2.0961721077398031
$\boldsymbol{b_0-b_2}$	650917	797207	1127421	1271080364935	2363237	2.0961423504846725
$\boldsymbol{b_0+b_2}$	480080	587976	831523	691432162129	1742929	2.0960657786947677
$\boldsymbol{b_0+b_1+b_2}$	402472	492926	697103	1485952593505	461153	2.0960360214386713
$\boldsymbol{b_0+b_1+2b_2}$	967971	1185518	1676576	32810907084622	514099	2.0959974373193818
$\boldsymbol{b_0+2b_1-b_2+b_3}$	719012	880607	1245366	1550936473956	2608466	2.0945376700504108
$\boldsymbol{b_0-b_1+b_2+b_3}$	644810	789727	1116843	21247338286649	339242	2.0945128366296784
$\boldsymbol{b_0+b_2+b_3}$	567202	694677	982422	965152986084	2057754	2.0945723935335324
$\boldsymbol{b_0+b_1+b_2+b_3}$	489594	599627	848001	719105696001	1776250	2.0946319638774010
$\boldsymbol{b_0+2b_1+b_2+b_3}$	411986	504577	713580	509196416400	1494730	2.0946915552565935
$\boldsymbol{b_0+2b_1+2b_2+b_3}$	977485	1197169	1693053	2866428460809	3546553	2.0947678542845380
$\boldsymbol{b_0+3b_1+2b_2+b_3}$	899877	1102119	1558632	2429333711424	3265065	2.0948273870932972

Table 4.2: Examples of magnitude about 10^{12} with large diameter (output of KASH-command DiamLarge (env, DiamTheta(env)-0.003, 3*10^11, 3*10^12, 10^9))

indicated in the preceding section, we cannot simply take $X = y^2$ in this case; the table includes the value of X found after a maximum lookup from the initial value y^2.

Extending these computations to larger values of n and k is not difficult. In Table 4.3, we list the output of our *small*-algorithm for X of all magnitudes from 10^3 up to 10^{30}, for the case $n = 4$, $k = 3$, $a_1 = 7$, $a_2 = 5$, $a_3 = 3$, $a_4 = 2$.[1] Taking two magnitudes at a time, the task is accomplished by the KASH-commands

```
> result := [ ];
> for i in [1..14] do
>   Append (result, DiamSmall (env, 4, 3*10^(2*i), 3*10^(2*i+2), 0));
> od;
> Sort (result, function (x,y) return (x[2]<y[2]); end);
```

The last command sorts the output by increasing X. Entering 0 instead of a valid magnification factor lets the DiamSmall function calculate an appropriate magnification factor; the formula we have used is given below. We have taken 4 (instead of $\psi \approx 11.7389$) as the upper bound for $\mathrm{diam}(X)/X^{5/9}$ to limit the size of the output.

Our strategy is again to look for appropriate $x_i \in \mathbb{Z}$ and then put $X = a_n x_n^k = 2x_4^3$. For $i = 1, 2, \ldots, 14$ we find the solutions with $3 \cdot 10^{2i} < X \leq 3 \cdot 10^{2i+2}$ by looking in the lattice defined by the matrix

$$\mathcal{A}_i = \begin{bmatrix} \mu_i & 0 & 0 & -[\mu_i \sqrt[3]{2/7}] \\ 0 & \mu_i & 0 & -[\mu_i \sqrt[3]{2/5}] \\ 0 & 0 & \mu_i & -[\mu_i \sqrt[3]{2/3}] \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where μ_i is the magnification factor for range i.

What number of lattice points will have to be investigated, and how does this depend on the magnification factor? If we want $X \leq U$ for some upper bound U, then we have $x_4 \leq \sqrt[3]{U/a_4}$. Furthermore, from a lower bound L for X, it follows that we want $x_4 \geq \sqrt[3]{L/a_4}$. However, the use of lattice techniques implies that it is easier to place restrictions on the *length* of lattice points. Therefore a suitable magnification factor should result in the same upper bound for the other components of interesting lattice points as we have for x_4. Our factor of choice will be $(U/a_4)^{1/k} \cdot (L/a_4)^{(1/k)\cdot(1/(n-1))}$. Thus μ_i is easily found to be approximately $5.56 \cdot 10^{8i/9}$.

We can now define a search for lattice elements based on their lengths: a lattice vector $\boldsymbol{x}$ should satisfy

$$\sqrt[3]{L/a_4} \leq \|\boldsymbol{x}\| \leq 4 \cdot \sqrt[3]{U/a_4}.$$

[1] We start at 10^3 because almost all $X \leq 300$ have $\mathrm{diam}(X) \leq \psi\, X^{5/9}$; 5/9 is the value of the 'low exponent' $1 - 1/k - 1/k(n-1)$ in this setting.

X	$\mathrm{diam}(X)$	$\mathrm{diam}(X)/X^{5/9}$
6750	409	3.050...
24334	940	3.438...
109744	1949	3.087...
1362944	10181	3.978...
17230250	24233	2.313...
588159250	209573	2.814...
1317006000	132395	1.136...
7233610750	934146	3.111...
10536048000	1059160	2.862...
27821160250	2450663	3.862...
14186764344688	63265187	3.122...
16202447905408	85772915	3.932...
1056360627067408	147492283	0.664...
8450885016539264	1179938264	1.673...
28521736930820016	3982291641	2.873...
80389978262750000	9513643375	3.859...
151786560772574208	10487193792	2.989...
256380524661296464	11147005161	2.374...
400510033691321216	11785971409	1.959...
590513251625052912	15179582750	2.034...
832728342224896000	26426874625	2.925...
1133493469253254928	41471038500	3.868...
257222405634587843114	872255518248	3.995...
269778872015666598366	412415638911	1.839...
282737464284192625762	626104350754	2.721...
389097405838209605994	1039968389952	3.785...
274899795914113414208512	39320333109789	3.739...
2242037388804686746669424	119146919065839	3.531...
2590762726016380940458250	130611041439248	3.572...
37042484784988791775041594	596513845397437	3.722...
104283277218850000159072784	447145317839800	1.569...
519802244616317526257559662	2697788013092818	3.880...
834266217750800001272582272	3577162542718400	3.956...
87458080363283196729963494822	34681620139888675	2.892...
118334934739697831603535146218	19968864177856453	1.408...
946679477917582652828281169744	159750913422851624	3.548...

Table 4.3: Examples of various magnitudes with $\mathrm{diam}(X)/X^{5/9} < 4$

Thus all points of interest are to be found in an 4-dimensional "annulus" with the given inner and outer radii. The volume of this annulus is given by

$$V_4(4 \cdot \sqrt[3]{U/a_4}) - V_4(\sqrt[3]{L/a_4})$$

where $V_4(r) = \frac{\pi^{4/2}}{\Gamma(\frac{1}{2}4+1)} r^4 = \frac{1}{2}\pi r^4$ is the 4-dimensional form of the general formula for the volume of an n-sphere [1]. Substituting the bounds given above, we find this volume to be approximately $62920 \cdot 10^{\frac{8i}{3}}$. Dividing by the discriminant of our lattice, which is clearly $\mu_i^3 \approx 172 \cdot 10^{\frac{8i}{3}}$, we find about 366 lattice points in our region, which is small, and, more importantly, independent of i. As the region is symmetric around $\mathbf{0}$, and KASH only generates one of $\boldsymbol{x}$ and $-\boldsymbol{x}$, we find that our search will involve no more than 200 lattice points per two magnitudes.

The 'best' example we thus found has got

$$\begin{aligned} x_1 &= 53240, & 7x_1^3 &= 1056360565568000, \\ x_2 &= 59559, & 5x_2^3 &= 1056360604069395, \\ x_3 &= 70615, & 3x_3^3 &= 1056360479575125, \\ x_4 &= 80834, & 2x_4^3 &= 1056360627067408, \end{aligned}$$

and

$$X = 1056360627067408, \quad \mathrm{diam}(X) = 147492283.$$

Note that $x_4 = 2 \times 80834 = 161668$ and $3 \times 80834 = 242502$ are also in our list (they correspond to the next two entries after $X = 1056360627067408$).

It is clear that increasing the magnification factor will result in less work, and less examples found. However, as the product of $\mathcal{A}_i$ and $\boldsymbol{x} = (x_1, x_2, x_3, x_4)$ grows linearly with μ_i, it is clear that the 'best' examples will eventually remain. For example, if we multiply μ_i by $10^{14/9} \approx 36$, KASH only investigates 10 lattice points in total for *all* magnitudes, while producing the 8 best examples of our list.

On the other hand, reducing μ_i to $3 \cdot 10^{8i/9}$ leads to the investigation of about $14 \times 1170 = 16380$ lattice points in total; this search produces 103 examples, but after removing duplicates we are left with the same 36 that are already on our list.

We may conclude that our value for the magnification factor seems to be a good compromise, at least in this particular setting, as it produces 'all' relevant examples without generating duplicates. This search takes KASH less than one minute on a (rather busy) SGI Indy with an R5000 processor.

Chapter 5

Lower bounds for the 'quality' of consecutive almost powers

5.1 Introduction

As indicated in the Introduction, we intend to find pairs (e, c), with $e \geq 0$ and $c > 0$ real constants, such that

$$\liminf_{m\to\infty} \frac{\lambda_{n,k}(m)}{m^e} \leq c, \tag{5.1}$$

where n and k are positive integers greater than 1. We have obtained the following results:

(i) Let $n \geq 2$ and $k \geq 2$. Then (5.1) is satisfied by $e = 1 - \frac{k}{nk-1}$ and some effectively computable constant $c = c(n, k)$.

(ii) If n and k are odd, we may take $e = 1 - \frac{k}{nk-2}$.

(iii) If $n = k = 2$, we have (5.1) for $e = 0$ and $c = 2$.

In the remaining sections, we will first give our proof of these claims and then pay attention to computational aspects of this problem.

5.1.1 Construction of families of 'small examples'

As we have remarked earlier, we may take $e = 1$ for any n and k. However, we can establish better solutions e to (5.1) by constructing increasing sequences of positive integers $(m_i)_{i=1}^{\infty}$ such that $\lambda_{n,k}(m_i) \leq cm_i^e$ for all $i \geq 1$ and some constant $c > 0$. This task is easy in case $n = k = 2$.

Lemma 5.1 *If $n = k = 2$, (5.1) holds with $e = 0$ and $c = 2$.*

Proof. As the Pell equation $x^2 - 2y^2 = 1$ has infinitely many integral solutions (x, y), we find an infinitude of positive integers m such that $\lambda_{2,2}(m) = 2$. □

For general n and k, the construction of sequences $(m_i)_{i=1}^{\infty}$ starts with the consideration of systems of linear congruences of the following form:

$$\begin{cases} m_i \equiv b_1 & \pmod{c_1(i)^k} \\ m_i \equiv b_2 & \pmod{c_2(i)^k} \\ \vdots & \\ m_i \equiv b_n & \pmod{c_n(i)^k} \end{cases}, \tag{5.2}$$

where the evaluations (as I and MapleV3 call them) $b_1, \dots, b_n$ form a sequence of consecutive integers, and the moduli $c_1^k, \dots, c_n^k$ are positive integers, depending on i. Now if m_i satisfies a system of this form for all i, we have

$$\lambda_{n,k}(m_i) \le \frac{m_i - b_1}{\min_{j=1\dots n} c_j(i)^k} \,. \tag{5.3}$$

Furthermore, using the Chinese Remainder Theorem and the fact that not all moduli are zero, it follows that we may assume

$$0 < m_i < \operatorname{lcm}(c_1(i)^k, \dots, c_n(i)^k) \le (c_1(i) \cdot \dots \cdot c_n(i))^k \,. \tag{5.4}$$

To construct our actual m_i, we resort to polynomial methods: from now on, we will assume that the $c_j(i)$ are obtained by substituting i into a polynomial $f \in \mathbb{Z}[x]$. It would clearly be sufficient if f were in $\mathbb{Q}[x]$ and $f(\mathbb{Z}) \subset \mathbb{Z}$, but we will not need that here.

We will make use of both assertion and proof of the Chinese Remainder Theorem for polynomials, which we adapt from the integer version given by Cohen [9, Corollary 1.3.10]. Furthermore, we need a little symmetry result.

Lemma 5.2 *Let $c_1(x), \dots, c_n(x)$ be mutually relatively prime polynomials over a field $\mathbf{K}$, and let $b_1(x), \dots, b_n(x)$ be arbitrary polynomials over $\mathbf{K}$. Then there exists a polynomial $f(x) \in \mathbf{K}$, with degree less than $\sum_{j=1}^n \deg c_j$, which satisfies*

$$\begin{cases} f(x) \equiv b_1(x) & \pmod{c_1(x)} \\ f(x) \equiv b_2(x) & \pmod{c_2(x)} \\ \vdots & \\ f(x) \equiv b_n(x) & \pmod{c_n(x)} \end{cases}, \tag{5.5}$$

and which is unique modulo $\prod_{j=1}^n c_j(x)$.

Proof of Lemma 5.5. Put $M_j(x) = \prod_{\ell=1,\ell\neq j}^n c_\ell(x)$, and let $y_j(x)$ be the polynomial inverse of $M_j(x)$ modulo $c_j(x)$. Put $f(x) = \sum_{j=1}^n b_j(x) y_j(x) M_j(x) \bmod \prod_{j=1}^n c_j(x)$. Then $f(x)$ satisfies our claims, which may be verified in a straightforward manner. □

Lemma 5.3 *Let $f(x)$ be any polynomial over a field $\mathbf{K}$, such that it has a polynomial inverse $g(x)$ modulo x^{m+1} for some nonnegative integer m. Then $f(-x)g(-x) \equiv 1 \pmod{x^{m+1}}$ as well.*

Proof of Lemma 5.3. If $x^{m+1} \mid f(x)g(x) - 1$, then $(-x)^{m+1} \mid f(-x)g(-x) - 1$. □

Theorem 5.4 *We may take $e = 1 - \frac{k}{nk-1}$ for $n, k \ge 2$. If, additionally, both n and k are odd, we may take $e = 1 - \frac{k}{nk-2}$.*

Proof of Theorem 5.4. We first prove the case when n is odd. The even case will be analogous.

Consider the system

$$\begin{cases} m_i \equiv -\frac{n-1}{2} & \pmod{(ai - \frac{n-1}{2})^k} \\ m_i \equiv -\frac{n-3}{2} & \pmod{(ai - \frac{n-3}{2})^k} \\ \vdots & \\ m_i \equiv \frac{n-1}{2} & \pmod{(ai + \frac{n-1}{2})^k} \end{cases} \tag{5.6}$$

where a is an integer parameter which will be assigned a value later on. Thus we have $c_j(x) = ax + j$, letting our index j run from $-\frac{n-1}{2}$ to $\frac{n-1}{2}$. Note that $m_i \equiv 0 \pmod{(ai)^k}$.

Following the proof of Lemma 5.2, we write

$$M_j(x) = \prod_{\ell=-\frac{n-1}{2},\ell\neq j}^{\frac{n-1}{2}} (ax+\ell)^k = \prod_{\ell=-\frac{n-1}{2},\ell\neq j}^{\frac{n-1}{2}} (ax+j+(\ell-j))^k \tag{5.7}$$

as a polynomial $\sum_{\ell=0}^{(n-1)k} m_{j,\ell}(ax+j)^\ell$ in $ax+j$, and $y_j(x)$ as $\sum_{\ell=0}^{k-1} y_{j,\ell}(ax+j)^\ell$. As $M_j y_j \equiv 1 \pmod{(ax+j)^k}$, we can solve the coefficients $y_{j,\ell}$ from the linear systems

$$\mathbf{A}_j\, \vec{y}_j =_{def} \begin{bmatrix} m_{j0} & 0 & \dots & 0 \\ m_{j1} & m_{j0} & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ m_{j,k-1} & m_{j,k-2} & \dots & m_{j0} \end{bmatrix} \begin{pmatrix} y_{j0} \\ y_{j1} \\ \vdots \\ y_{j,k-1} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

for $j = -\frac{n-1}{2}, \dots, \frac{n-1}{2}$. We have $\det \mathbf{A}_j = m_{j0}^k$. From (5.7), we have $m_{j0} = \pm\left(\frac{n-1}{2}+j\right)!^k \cdot \left(\frac{n-1}{2}-j\right)!^k$; as the binomial coefficient $\binom{n-1}{\frac{n-1}{2}+j}$ is integral, we find that $m_{j0} \mid (n-1)!^k$. Thus $(n-1)!^{k^2} y_j(x) \in \mathbb{Z}[x]$, for all j.

Let us denote by $\operatorname{cont} P$ the *content* of a polynomial $P \in \mathbb{Z}[x]$, i.e. the greatest common divisor of its coefficients. We have $f(x) = \sum_{j=-\frac{n-1}{2}}^{\frac{n-1}{2}} j y_j(x) M_j(x)$. As $a^k \mid \operatorname{cont}(M_j)$ for $j \neq 0$, and M_0 cancels out, $f(x)$ has integer coefficients if we choose a equal to $(n-1)!^k$.

It remains to investigate the quotients of $f(x)-j$ and $(ax+j)^k$, for $j = -\frac{n-1}{2}, -\frac{n-3}{2}, \dots, \frac{n-1}{2}$, which we know to be in $\mathbb{Q}[x]$. It is an implication of Gauss' Lemma that if $g, h \in \mathbb{Z}[x]$ with $g \mid h$, and g has content one, then $h/g \in \mathbb{Z}[x]$. We may thus conclude that $f(x)-j$ is divisible over $\mathbb{Z}$ by $((a/j)x+1)^k$ if $j \neq 0$, and by x^k if $j=0$, recalling that $j \mid a$ for all j considered, and that the equations (5.5) remain valid upon multiplying some $c_j(x)$ by a scalar.

We take our solution m_i to be $f(i)$, so that

$$\lambda_{n,k}(m_i + \frac{n-1}{2}) \leq \frac{m_i}{i^k},$$

an upper bound which is a polynomial in i of degree $d \leq (n-1)k-1$. It follows that $e = \frac{d}{d+k} = 1 - \frac{k}{d+k} \leq 1 - \frac{k}{nk-1}$.

We can, in some cases, improve this bound as follows. Suppose k is odd. It is not difficult to see that $M_{-j}(x) = M_j(-x)$. Thus by our second Lemma, we also have $y_{-j}(x) = y_j(-x)$. We find that the leading coefficient of $M_{-j}y_{-j}$ is $m_{-j,(n-1)k}y_{-j,k-1} = (-1)^{n(k-1)} m_{j,(n-1)k} y_{j,k-1} = m_{j,(n-1)k} y_{j,k-1}$, as k is odd. Thus the leading term of $jy_jM_j - jy_{-j}M_{-j}$ cancels out, yielding an $f(i)$ of degree $d \leq (n-1)k-2$, and a 'solution' e of $\frac{d}{d+k} \leq 1 - \frac{k}{nk-2}$ in this case. If $n \equiv 3 \pmod 4$, it appears that our solution sequence m_i consists of negative numbers; this can be amended by reversing the order of the moduli, or of the numbers $-\frac{n-1}{2}, -\frac{n-3}{2}, \dots, \frac{n-1}{2}$.

Finally, when n is even, we consider

$$\begin{cases} m_i \equiv 0 & \pmod{(ai)^k} \\ m_i \equiv 1 & \pmod{(ai+1)^k} \\ \vdots & \\ m_i \equiv n-1 & \pmod{(ai+n-1)^k} \end{cases}.$$

The proof runs entirely analogously, except that j runs from 0 to $n-1$, and the symmetry considerations for odd k cannot be extended to this case. Note that the same value for a suffices to obtain integral solutions; we have $m_{j0} = \pm j!^k \cdot (n-1-j)!^k \mid (n-1)!^k$. $\square$

5.1.2 Computational aspects

How do our bounds perform?

To give an impression of what our bounds amount to, we have included some pictures (Figures 5.1 through 5.3) showing $\frac{\log \lambda_{n,k}(X)}{\log X}$ for all $X < 1{,}000{,}000{,}000$ with reasonably small $\lambda_{n,k}(X)$, for some values of n and k (X runs up to 100,000,000 if $n = 2$). Taking this logarithm ratio has the advantage that our alleged lower bounds take the form of straight horizontal lines (the dashed lines in our graphs). Eventually, if X grows large enough, all generated points should lie above these lines.

We have computed these values using the C++-programs listed in Appendix C. Obviously, the number of interesting examples decreases dramatically with k. If $k \geq 4$, there is no X below $1,000,000,000$ for which $\lambda_{n,k}(X)$ is reasonably small (outside the truncation zone in our pictures).

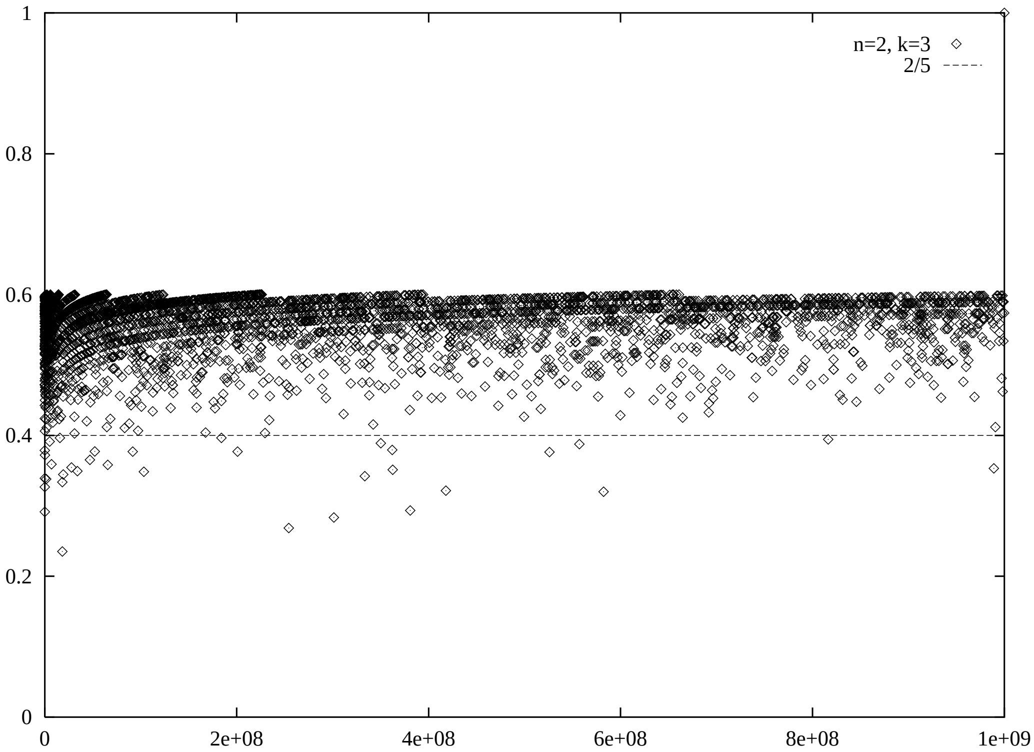

Figure 5.1: $\log \lambda_{2,3}(X)/\log X$ (for all X with this ratio < 0.6)

We can conclude that our lower bounds for e seem reasonable for the shown values of $k = 3$. Some more experiments with $k = 2$ suggest that it should be possible to prove something better in this situation.

Implementation in MapleV

We have included a transcript of a MapleV, Release 3, session that may be used to evaluate the polynomials discussed in the proof of Theorem 5.4. The MapleV code for the polynomial CRT routine may be found in the Appendix, p. 44; the code below supposes this routine to be available in the (text) file polcrt.script. The code is suitable for any value of n and k, though to give a concrete illustration we have taken $n = k = 3$.

The moduli function returns the 'reduced' moduli, i.e. those with trailing coefficient 1.

It should be noted that the choice $a = (n-1)!^k$ is an upper bound, and is much too high in many concrete cases. For fixed n and k, one may establish a tighter upper bound by reducing the linear system in $\mathbf{A}_j$ and $\vec{y}_j$. As most equations are homogeneous, one might try to divide out the gcd of a row of entries in $\mathbf{A}_j$, or to increase this gcd by performing row additions, thus reducing $\det \mathbf{A}_j$. E.g., for $n = 9$ and $k = 4$, $a = 5880 = 2^3 \cdot 3 \cdot 5 \cdot 7^2$ is enough to obtain an integral expression for m_i, whereas $8!^4 = 2642908293365760000$; for $n = 16$ and $k = 4$, $a = 360360 = 2^3 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11 \cdot 13$ is sufficient, whereas $(16-1)!^4 \approx 10^{24}$.

When evaluating a concrete case, we recommend to take as initial value for a the product of all primes dividing $(n-1)!$, and adding some factors as necessary (these may be derived from the denominators appearing in the output of the Chinese Remainder Theorem, and especially the denominator of the constant term). In the script below, a is initialized to a multiple of this value, namely, the least common multiple of $1, 2, \ldots, n-1$.

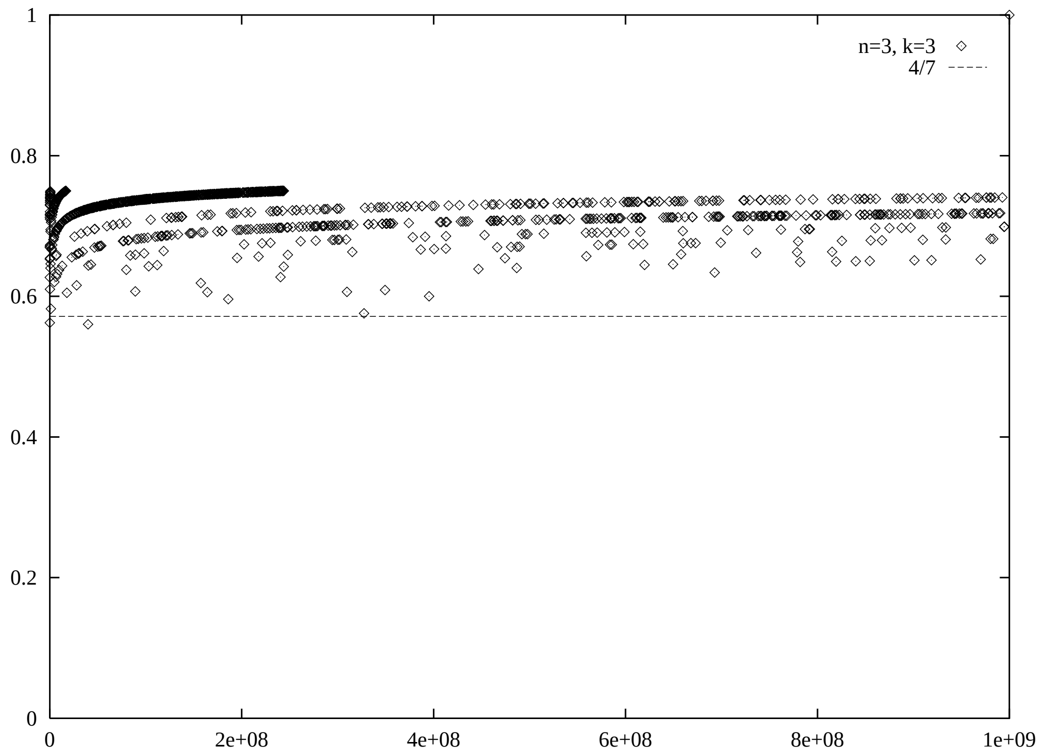

Figure 5.2: $\log \lambda_{3,3}(X)/\log X$ (for all X with this ratio < 0.75)

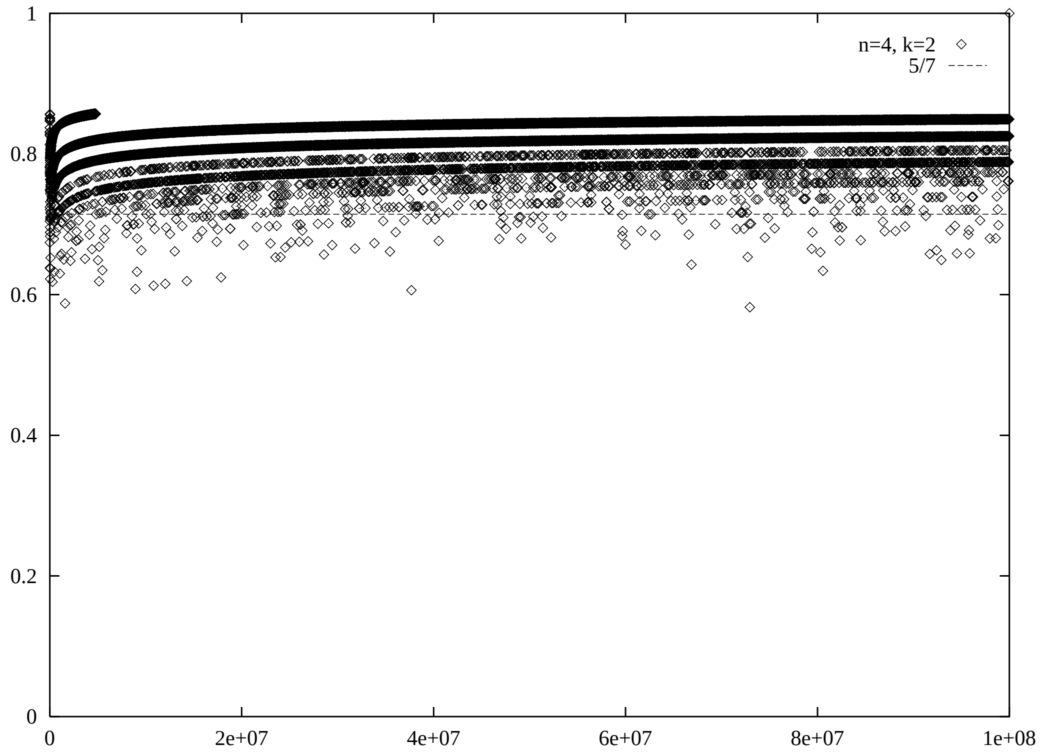

Figure 5.3: $\log \lambda_{4,2}(X)/\log X$ (for all X with this ratio < 0.857)

```
> read ( `polcrt.script` );          # Needed for polcrt routine
```

```
> evaluations := proc (n)
>   local i;
>   if type (n, odd) then
>     if irem (n, 4) = 3 and type(k, odd) then
>       RETURN ( [ seq ( -i, i = -(n-1)/2..(n-1)/2 ) ] );
>     else
>       RETURN ( [ seq ( i, i = -(n-1)/2..(n-1)/2 ) ] );
>     fi
>   else
>     RETURN ( [ seq ( i, i = 0..n-1 ) ] );
>   fi;
> end:
```

```
> moduli := proc (n, k, a)
>   local i;
>   if type (n, odd) then
>     RETURN ( [ seq ( (a/i * x + 1)^k, i = -(n-1)/2..-1 ), (a*x)^k,
>                seq ( (a/i * x + 1)^k, i = 1..(n-1)/2 ) ] );
>   else
>     RETURN ( [ (a*x)^k, seq ( (a/i * x + 1)^k, i = 1..n-1 ) ] );
>   fi;
> end:
```

```
> n := 3; k := 3; a := ilcm ( seq ( i, i=1..n-1 ) );
```

$$n := 3$$

$$k := 3$$

$$a := 2$$

```
> evaluations (n);
```

$$[1, 0, -1]$$

```
> moduli (n,k,a);
```

$$\left[(-2\,x+1)^3, 8\,x^3, (2\,x+1)^3\right]$$

```
> p := sort ( simplify ( polcrt ( evaluations (n), moduli (n,k,a), x ) ) );
```

$$p := 240\,x^7 - 168\,x^5 + 35\,x^3$$

```
> p5 := subs(x=5,p);
```

$$p5 := 18229375$$

```
> ifactor ( [ p5-1, p5, p5+1 ] );
```

$$\left[(2)\,(3)^6\,(12503), (5)^4\,(29167), (2)^7\,(11)^3\,(107)\right]$$

As may be seen, taking $a = 2$ and evaluating the system of linear congruences (5.6) results in a polynomial $p(x)$ of degree 7 with integer coefficients. On examining $p(5)$, e.g. one finds that $9^3 \mid p(5) - 1$, $5^3 \mid p(5)$ and $11^3 \mid p(5) + 1$, as claimed.

5.2 Is our bound sharp?

Just as in the diameter case, we try to answer the question whether the constructed possible values for e are optimal. Again, there are deep methods involved in getting anything out. What we would want to prove is, ultimately, that now we have indeed characterized all values of e which satisfy (5.1) — that is to say: that, for any $\varepsilon > 0$ and any real constant $c > 0$, there are but a finite number of (positive) integers X satisfying

$$\lambda_{n,k}(X) < c \cdot X^{1-\frac{k}{nk-1}-\varepsilon}$$

or, in case $n \cdot k$ is odd,

$$\lambda_{n,k}(X) < c \cdot X^{1-\frac{k}{nk-2}-\varepsilon}.$$

In the mean time, we are far from proving anything of the sort: the results we will discuss in the following sections are either polynomial in $\log X$, or depend on an unproved conjecture (the so-called *abc*-conjecture). The quoted result of Turk (Theorem 6 in Chapter 1), which is of the first sort, goes without further comments, except noting that Turk's proof, being based on the theory of linear forms in logarithms, is effective.

5.2.1 The Thue equation

We will view the problem of finding $X \in \mathbb{N}$ where all of $X, X+1, \ldots, X+n-1$ contain reasonably large kth power factors, as the study of the system of Thue equations

$$\begin{cases} a_2x_2^k - a_1x_1^k &= 1 \\ a_3x_3^k - a_1x_1^k &= 2 \\ &\vdots \\ a_nx_n^k - a_1x_1^k &= n-1 \end{cases},$$

which is clearly solvable if we substitute the kth power-free part of the consecutive integers $X, X+1, \ldots, X+n-1$ for the coefficients $a_1, \ldots, a_n$ (thus we have $X = a_1x_1^k$).

We will make use of a result of Bugeaud and Győry, which estimates the size of the solutions of a Thue equation, i.e. an equation of the form $F(x,y) = b$ in $x, y \in \mathbb{Z}$, where $F(x,y)$ is a bivariate polynomial of degree $k \geq 2$ with integer coefficients. Their Theorem reads [7, Theorem 3]:

Lemma 5.5 *Let (x,y) be a solution of $F(x,y) = b$, as described above. Then*

$$\max\{|x|,|y|\} < \exp\{c_4H^{2k-2}(\log H)^{2k-1}\log B\}, \tag{5.8}$$

where $B \geq 3$ is an upper bound for b, $k \geq 3$ is the degree of the equation, $H \geq 3$ is an upper bound for the height of F, i.e. the maximum absolute value of its coefficients, and $c_4 = c_4(k) = 3^{3(k+9)}k^{18(k+1)}$.

Note that the case $k = 2$ is not covered by this result.

Assume now that $n \geq 2$, take some $X \in \mathbb{N}$, $X > 2$, and consider the equation

$$a_2x_2^k - a_1x_1^k = 1,$$

taking the coefficients a_2 and a_1 as defined above. Thus we may take $B = 3$. On putting $X = a_1x_1^k$ and $H = \max\{3, |a_2|, |a_1|\}$, it follows that

$$\lambda_{n,k}(X) \geq H$$

, unless both coefficients are less than 3 in absolute value. However, this would constitute a nontrivial solution to the Diophantine equation $|x^k - 2y^k| = 1$, and we have

Lemma 5.6 *The Diophantine equation $|x^k - 2y^k| = 1$ has no solutions with $|xy| > 1$.*

Proof of S. uppose that $2y^k - x^k = 1$ for integers x, y. Then $(x, 1, y)$ is a solution to the equation

$$x^k + y^k = 2z^k. \tag{5.9}$$

Suppose further that $x^k - 2y^k = 1$, and that k has an odd prime factor ℓ. Then $(x^{k/\ell}, -1, y^{k/\ell})$ solves (5.9) with k replaced by ℓ. However, Darmon and Merel have proved [10] that any solution (x, y, z) in integers of (5.9) has either $|xyz| \leq 1$ or $\gcd(x, y, z) > 1$.

As a last case, let $x^k - 2y^k = 1$, and suppose that $k = 2^\ell$ for $\ell > 1$. Now we may invoke a theorem of Skolem [12, Theorem 7 of Chapter 23] which states that the only integer solutions of $x^4 - 2y^4 = 1$ are given by $y = 0$, $x = \pm 1$. □

We have

$$\log X \leq \log(\max\{|x|, |y|\}^k \cdot H) < k \cdot c_4 \cdot H^{2k-2} (\log H)^{2k-1} \log B + \log H$$

so that, for some positive c_5,

$$c_5 \cdot (\log X)^{\frac{1}{2k-2}} < H (\log H)^{\frac{2k-1}{2k-2}} \leq \left(\frac{2k-1}{e} H \right)^{\frac{2k-1}{2k-2}}.$$

We have thus proved the following theorem:

Theorem 5.7 *Let $n \geq 3$ and $k \geq 3$. Then for all X we have*

$$\lambda_{n,k}(X) > c_6 \cdot (\log X)^{\frac{1}{2k-1}}$$

for a certain positive constant c_6.

5.2.2 The *abc*-conjecture

In this section we will explore the consequences of the *abc*-conjecture for the existence of consecutive integers with large perfect power divisors. We denote by $r(n)$ the radical of a number $n \in \mathbb{Z}$, i.e. the product of all distinct primes dividing n.

The *abc*-conjecture, stated by J. Oesterlé and D. Masser, runs as follows [11, 6]:

Conjecture 5.8 *For all $\varepsilon > 0$, there exists a positive constant C_ε, such that for any integers $a, b, c > 0$ satisfying $a + b = c$ and $\gcd(a, b) = 1$, we have*

$$r(abc) > C_\varepsilon \cdot c^{1-\varepsilon}.$$

This clearly implies that for any integers $0 < x < y$ we have

$$(y - x) r(xy) > C_\varepsilon \cdot y^{1-\varepsilon}.$$

It is possible to apply this conjecture to our problem, and we will obtain nearly optimal results, as the following theorem shows.

Theorem 5.9 *The abc-conjecture implies that (5.1) cannot be solved if*

$$e < 1 - \frac{k}{nk - n}. \tag{5.10}$$

We derive this theorem from

Proposition 5.10 *Let n and $k \in \mathbb{Z}$, both greater than 2, and let $\delta > 0$. For any $X \in \mathbb{N}$ write*

$$X = a_1 x_1^k, X+1 = a_2 x_2^k, \ldots, X+(n-1) = a_n x_n^k,$$

with the a_i k-free. Then we have

$$a_1 a_2 \cdots a_n > X^{\frac{(n-1)k-n-\delta}{k-1}}$$

for all sufficiently large X.

For the proof we make use of a theorem of M. Langevin [11, Théorème 2], which we state as a lemma.

Lemma 5.11 (Langevin) *Let $b_1 < b_2 < \ldots < b_n$ be integers and $\varepsilon > 0$ in $\mathbb{R}$. The abc-conjecture implies*

$$r((X+b_1)(X+b_2)\cdots(X+b_n)) > X^{n-1-\varepsilon} \text{ for all sufficiently large } X. \tag{5.11}$$

As Langevin's own proof of this theorem is rather concise, and appears to be uncomplete as well, we present a more worked out version in Appendix D.

Proof of Proposition 5.10. Take n and $k \in \mathbb{Z}$, both greater than 2, and take $\varepsilon > 0$ small enough. Putting $b_1 = 0, \ldots, b_n = n-1$, we have, by Lemma 5.11,

$$X^{n-1-\varepsilon} < r(X(X+1)\cdots(X+(n-1))) \leq (a_1 a_2 \cdots a_n) \left(\frac{(X+n)^n}{a_1 a_2 \cdots a_n} \right)^{1/k},$$

so that

$$(a_1 a_2 \cdots a_n)^{1-1/k} > X^{n-\frac{n}{k}-1-\varepsilon} \cdot (1+\varepsilon')^{-\frac{n}{k}} \geq X^{n-\frac{n}{k}-1-\delta}$$

for some $\varepsilon' > 0$. This proves the Proposition. □

Proof of Proof of Theorem 5.9.. From our Proposition, it follows that

$$\max\{a_1, a_2, \ldots, a_n\} > X^{\frac{(n-1)k-n-\delta}{nk-n}}$$

for any $\delta > 0$ and all sufficiently large $X \in \mathbb{N}$. Our Theorem follows. □

Browkin and Brzeziński have provided a generalization of the abc-conjecture, which they call the n-conjecture (cf. [6]). There, they deal with n integers summing up to zero, and the exponent $1-\varepsilon$ is replaced by $2n-5-\varepsilon$. It might be interesting to try and generalize Langevin's results to cover this more general case, and see whether this provides a better bound for e.

Appendix A

KASH-sources

A.1 Calculations with diameters

The following KASH-source file implements diverse features which could facilitate computations with diameters, among which are functions to find *some* examples of large c.q. small diameter within a given range of integers. More explanation may be found in the source code itself. The following functions are provided:

env := DiamEnvironment (n, k, [a_1, ..., a_n]): Defining a diameter environment.

Diam (X, env): Calculating a diameter. Returns a list containing both the actual diameter value, and a list of the x_i^*.

DiamTheta (env): Calculating ϑ.

DiamPsi (env): Calculating ψ.

DiamHiExp (env): Calculating $1 - 1/k$.

DiamHiratio (env, X): Calculating $\mathrm{diam}(X)/X^{1-1/k}$.

DiamLowExp (env): Calculating $1 - 1/k - 1/k(n-1)$.

DiamLowratio (env, X): Calculating $\mathrm{diam}(X)/X^{1-1/k-1/k(n-1)}$.

DiamSmall (env, maxratio, lower, upper, magn): Finding examples X with lower $\leq X \leq$ upper and DiamLowratio (env, X) $\leq$ maxratio. magn defines the granularity of the search: a small magn means fast searching and few examples, a large magn means the opposite. One may supply 0 to let the system define an appropriate value.

DiamMagnSmall (env, lower, upper): Finding the appropriate magnification factor just referred to.

DiamLarge (env, minratio, lower, upper, magn): Finding examples with large diameter, analogously, except for an additional maximum lookup.

DiamMagnLarge (env, lower, upper): Analogous to DiamMagnSmall.

DiamGammaMat (size): Finding the matrix associated to (2.7), which clearly only depends on m.

DiamGammaVec (env, magn): Calculating the reference vector used for finding lattice points associated to large diameter examples. Setting magn = 1 gives the γ_s.

DiamDioSystem (line, Gamma, env): Examining the compliance of found examples to the given Diophantine approximation systems.

```
# This is a KASH script, written 12/02/98 by Christiaan van de Woestijne
# as a MapleV3 script, translated to KASH and extended 11/06/98

# This script refers to CvdW's Master's Thesis for definitions etc. It makes
# use of a data structure called a 'diameter environment' and defined as
# a KASH record consisting of
#      n    an integer >= 3 (the number of almost powers needed)
#      k    an integer >= 3 (the exponent of the powers)
#      a    an integer list of dimension n, containing the 'weights' of the
#            almost powers
#      DoPrint   a boolean indicating whether some intermediate results in
#                the calculation of diameters should be printed
#      a_roots   the k-th roots of the values of a; by storing these, one
#                may speed up diameter calculations

# The function DiamEnvironment returns valid diameter environments from the
# n, k and a arguments; the a's are sorted in descending order.

DiamEnvironment := function (n_i, k_i, a_i)
  local i,b,roots;

  if (IsInt (n_i) and IsInt (k_i) and IsList (a_i)) then
    b := n_i >= 2 and k_i >= 2 and Length(a_i) = n_i;
    for i in [1..Length(a_i)] do
      if not IsInt (a_i[i]) then
        b := false;
      fi;
    od;
  else
    b := false;
  fi;
  if b then
    Sort (a_i);
    return ( rec (n:= n_i, k := k_i, a := Reversed (a_i), DoPrint := false) );
  else
    Print("This function requires an int n >= 2, an int k >= 2, and ",
          "a list a of positive integers such that their k-th roots ",
          "are linearly independent over Z.\n");
    return;
  fi;
end;

# The DiamPrecomp function performs the precomputations for diameter
# calculations, viz. the initialization of the a_roots array. Is called
# automatically by Diam if needed.

DiamPrecomp := function (env)
  local i;

  env.a_roots := [ ];
  for i in [1..env.n] do
    env.a_roots [i] := env.a[i]^(1/env.k);
  od;
end;

# Follows the actual diameter calculation function. Calling sequence:
#
```

```
#    Diam(X,env)

# Parameters:
#    X    a positive integer
#    env  a 'diameter environment'

# This function returns the almost-power diameter of a positive integer X,
# relative to the environment parameters listed above.

# It returns a list consisting of two elements: first, the actual diameter;
# second, a list containing the values of the x^*_i for which the diameter
# value is taken on.

# To clarify the thoughts a bit, let us write d for Diam (X)[1] and x for
# Diam(X)[2]. Then after completion of the function we have
#   d >= max ( seq ( seq ( abs (a[i]*x[i]^k - a[j]*x[j]^k), j=i+1..n ),
#                          i=1..n ) )
# i.e., if we take the maximal difference of two terms of the form a[i]*x[i]^k,
# we have almost got the diameter itself; the only catch is that we require X
# to be in the set { a[i]*x[i]^k | i=1..n }, which may increase the size of
# the final set, and d may be greater than this maximal difference.
# To be really complete, one has
#   d = max ( seq ( seq ( abs (a[i]*x[i]^k - a[j]*x[j]^k), j=i+1..n ),
#                         i=1..n ),
#                   seq ( abs (a[i]*x[i]^k - X), i=1..n ) ).

Diam := function (X, env)
  local x, l, r, xx,      # The l and r arrays feature in my proof of de
                          # Weger's theorem 1 about diameters
        i, j, cnt,
        hipow, prelimdiam;

  if not IsBound (env.a_roots) then
    DiamPrecomp (env);
  fi;

  # Let us first calculate  Floor ( ( X/env.a[i] )^(1/env.k) )

  x := [];
  l := [];
  r := [];
  xx := X^(1/env.k);
  for i in [1..env.n] do
    x [i] := Floor (xx / env.a_roots[i]);
    l [i] := [ env.a[i] * x[i]^env.k, i];
    r [i] := [ env.a[i] * (x[i]+1)^env.k, i];
  od;
  Sort (l);
  Sort (r);
  if env.DoPrint then
    Print ("X: ", X, "; Left: [");
    for i in [1..env.n-1] do
      Print (l[i][2], ", ");
    od;
    Print (l[env.n][2], "]; Right: [");
    for i in [1..env.n-1] do
      Print (r[i][2], ", ");
```

```
    od;
    Print (r[env.n][2], "]\n");
  fi;

  xx := [];
  prelimdiam := r[env.n][1] - l[1][1];                  # Initialize
  i := 1;
  j := 1;
  while i<env.n do                      # Find minimal index subsets
    i := i+1;                       # Disturb (and proceed...)
    while r[j][2] <> l[i-1][2] do   # Upper bound, use Lemma
      j := j+1;
    od;
    if j < env.n then
#      if PRINTLEVEL > 3 then
#        Print ("Right Index: ",j,"\n");
#      fi;

      # We know that our index subset is not minimal for any j smaller than
      # the one we have now. But we do not know yet whether i should be
      # increased to eliminate double occurrences of indices. Therefore, we
      # compare all 'right' indices to l[i][2], the leftmost bound of our
      # subset; if one is equal, then we should increase i and start again.
      # If cnt reaches zero, then we cannot increase our leftmost bound any
      # more without losing at least one index.

      cnt := j-1;
      while cnt > 0 do              # Lower bound, exhaustive search
        if r[cnt][2] <> l[i][2] then
          cnt := cnt - 1;
        else
          i := i+1;
          cnt := j;
        fi;
      od;
#      if PRINTLEVEL > 3 then
#        Print ("Left index: ",i,"\n");
#      fi;

      if r[j][1] - l[i][1] < prelimdiam then   # Found one, evaluate it!
        prelimdiam := r[j][1] - l[i][1];
#        Print ("Changed diam to ", prelimdiam, " and xx from ", xx);
        for cnt in [i..env.n] do               # Store x_i's used
          xx[l[cnt][2]] := x[l[cnt][2]];
        od;
        for cnt in [1..j] do
          xx[r[cnt][2]] := x[r[cnt][2]]+1;
        od;
#        Print (" to ", xx, "\n");
      fi;
    else
      i:=env.n;
    fi;
  od;

# There are two possibilities for evaluating the extremal minimal subsets.
# One demands that X itself should be in the set, the other allows the set
```

```
# to stretch out outside X. I choose the former one.

#  if l[env.n][1] - l[1][1] < prelimdiam then  # Evaluate extremal minimal subsets
  if X - l[1][1] < prelimdiam then
#    prelimdiam := l[env.n][1] - l[1][1];
    prelimdiam := X - l[1][1];
#    Print ("Changed diam to ", prelimdiam, " (left) and xx from ", xx);
    for cnt in [1..env.n] do
      xx[l[cnt][2]] := x[l[cnt][2]];
    od;
#    Print (" to ", xx, "\n");
  fi;
#  if r[env.n][1] - r[1][1] < prelimdiam then
  if r[env.n][1] - X < prelimdiam then
#    prelimdiam := r[env.n][1] - r[1][1];
    prelimdiam := r[env.n][1] - X;
#    Print ("Changed diam to ", prelimdiam, " (right) and xx from ", xx);
    for cnt in [1..env.n] do
      xx[r[cnt][2]] := x[r[cnt][2]]+1;
    od;
#    Print (" to ", xx, "\n");
  fi;

  return ( [prelimdiam,xx] );              # Ready
end;

# The following six functions perform some simple calculations with
# environments that should speak for themselves

DiamHiratio := function (X, env)
  local d;

  d := Diam (X, env)[1];
  return ( d / X^(1-1/env.k) );
end;

DiamHiExp := function (env)
  return (1-1/env.k);
end;

DiamLowExp := function (env)
  return (1-1/env.k-1/(env.k*(env.n-1)));
end;

DiamLowratio := function (X, env)
  local d;

  d := Diam (X, env)[1];
  return ( d / X^(1-1/env.k-1/(env.k*(env.n-1))) );
end;

DiamTheta := function (env)
  # Assume env.a is sorted in decreasing order

  return ( env.k * Maximum ( List ( [1..env.n],
           x -> Sum ( env.a{[1..x]}, y -> y^(1/env.k) ) / (x+1) ) ) );
end;
```

```
DiamPsi := function (env)
  # Assume env.a is sorted in decreasing order

  return ( env.k * env.a[env.n]^(1/(env.k*(env.n-1))) *
           (env.a[1]^(1/env.k) + env.a[2]^(1/env.k)) );
end;

# The last portion of this file addresses the problem of finding concrete
# examples of large or small diameter in a given range of integers.
# See the Examples section for explanation. The only extra feature is the
# choice of the magnification factor magn; if the DiamSmall or DiamLarge
# functions are called with a zero magn value, a calculated value is
# supplied that depends on both upper and lower.

# DiamLattice returns the lattice used for the calculations

# DiamMagnSmall and DiamMagnLarge return the magnification factors that would
# be used if the same arguments would be supplied to DiamSmall and DiamLarge,
# respectively

# DiamSmall returns *some* of the X with lower <= X <= upper and
# diam(X) <= maxratio * X^DiamLowExp (env). A reasonable first choice is
# maxratio = DiamPsi(env).
# DiamSmall returns a list of lists, each of which consists of
# - the representation of the associated lattice element w.r.t. the LLL-
#   reduced basis of it obtained by LatLLL
# - X itself
# - Diam (X)[2], i.e. the list of the x^*_i
# - Diam (X)[1], i.e. the actual diameter value of X
# - Diam (X)[1] / X^DiamLowExp (env), i.e. the ratio to be compared with
#   maxratio

# DiamLarge returns *some* of the X with lower <= X <= upper and
# diam(X) >= minratio * X^DiamHiExp (env). As DiamTheta (env) is really the
# maximal ratio, one might try DiamTheta (env) - 0.5, or something like that.
# DiamLarge performs an additional maximum lookup, as indicated in the main
# text.
# The function makes use of two additional functions, namely
#   DiamGammaMat (size), which returns the matrix associated with the linear
#     system which the \gamma_s values satisfy (it will be clear that this
#     matrix only depends on m and not on the a_i or whatever), and
#   DiamGammaVec (env, magn), which evaluates the \gamma_s values, multiplies
#     the resulting vector by magn and rounds off the resulting components
# A record of the function result consists of
# - the representation of the difference of the associated lattice element and
#   the reference vector, i.e. the lattice element closest to DiamGammaVec
#   (env, magn) according to LatShortestElt
# - X itself
# - Diam(X)[2]
# - Diam(X)[1]
# - Diam (X)[1] / X^DiamHiExp (env)

# Finally, DiamDioSystem takes one record ('line') of the output of one of the
# above functions and evalutes the quantities involved in the (in)homogeneous
# Diophantine approximation systems introduced in the main text. Thus the
# user may verify that the system needs (not) to be verified in order to
```

```
# find examples of sufficiently large or small diameter.

# A word about the implementation of matrix calculations below.
# There is an annoying bug in KASH, which results in the impossibility of
# addressing any matrix element in the rightmost column or bottom row.
# I work around this by first creating a matrix which has one more row
# and one more column than needed, and throwing these extra entries away
# after the completion of the computations. The following function may be
# used for the throwing away. It needs, apart from the matrix A to be cut,
# the order O from which A's coefficients are taken.

MatDecreasedSize := function (O, A)
  local B,i;

  B := MatToRowList (A);
  for i in [1..MatRows(A)-1] do
    B[i] := B[i]{[1..MatCols(A)-1]};
  od;
  return (Mat (O, B{[1..MatRows(A)-1]}));
end;

DiamLattice := function (env, magn)
  local A, i, j, B;

  A := MatId (Z, env.n+1);      # One too large, will be corrected later
  for i in [1..env.n-1] do
    A[i][i] := magn;
    A[i][env.n] := - Round (magn * ((env.a[env.n]+.0)/env.a[i])^(1/env.k) );
  od;
  A[env.n][env.n] := 1;
  return (Lat (MatDecreasedSize (Z, A)));

# A := MatId (R, env.n+1);
# for i in [1..env.n-1] do
#   A[i][i] := magn;
#   A[i][env.n] := - (magn * ((env.a[env.n]+.0)/env.a[i])^(1/env.k) );
# od;
# A[env.n][env.n] := 1.0;
# return (Lat (MatDecreasedSize (R, A)));
end;

DiamMagnSmall := function (env, lower, upper)

  # Code extracted from DiamSmall

  lower := Ceil ( (lower/env.a[env.n])^(1/env.k) );
  upper := Floor ( (upper/env.a[env.n])^(1/env.k) );
  return  (Ceil ( upper * lower^(1/(env.n-1))));
end;

DiamSmall := function (env, maxratio, lower, upper, magn)
  local L, expo, chi, phi, X, b, i, diam, result, toobig, toosmall,
        total;

  if (upper+.0)/lower > 10^(env.k-1) then
    Print ("WARNING: bounds differ by more than ", env.k-1, " magnitude(s)\n");
  fi;
```

```
  lower := Ceil ( (lower/env.a[env.n])^(1/env.k) );
  upper := Floor ( (upper/env.a[env.n])^(1/env.k) );
  if magn=0 then
    magn := Ceil ( upper * lower^(1/(env.n-1)) );
    # Compute the arithmetic mean of lower and upper. Next, we know that if
    # x_n > lower, then the minimal value of |x_i - (a_n/a_i)^(1/k)x_n| is
    # approximately lower^(-1/(n-1)).
  fi;

  Print ("Values: lower: ", lower, " upper: ", upper, " magn: ", magn, "\n");
  L := LatLLL (DiamLattice (env, magn));
  LatEnumReset (L);
  LatEnumLowerBound (L, lower^2);
  LatEnumUpperBound (L, env.n * upper^2 );
#  Print ("upperbound: ", env.n*upper^2, " lattice: ", L);

 # Comments on enumeration bounds: we require the last component to be >=
 # lower, and all components of our vector to be no larger than upper

  expo := DiamLowExp (env);
  toobig := 0;
  toosmall := 0;
  total := 0;
  result := [];
  while LatEnum (L) do
    chi := LatEnumElt (L);
    phi := LatEltVec (chi);
    Print ("Lattice element: ", chi);
    total := total+1;
    if phi[env.n][1] < 0 then phi := -phi; fi;

#   b := phi[1][1] < upper;
#   for i in [2..env.n] do
#     b := b and (phi[i][1] < upper);
#   od;
#   if b and lower <= phi[env.n][1] then
#     Print ((LatEltLength (chi)+.0) / (env.n*upper^2), " ", chi);

    if lower > phi[env.n][1] then toosmall := toosmall+1; fi;
    if upper <= phi[env.n][1] then toobig := toobig+1; fi;
    if lower <= phi[env.n][1] and phi[env.n][1] < upper then
      X := env.a[env.n] * phi[env.n][1]^env.k;
      diam := Diam (X, env);
      if diam[1] < maxratio * X^expo then
        Print (X, " has minimal diameter ", diam[1], ".\n");
        Add (result, [ LatEltToList (chi),  X,
                       diam[2], diam[1], diam[1]/X^expo ]);
      fi;
    fi;
  od;
# Sort (result, function (x,y) return (x[2] < y[2]); end);
  Print (Length (result), " small diam exs, ", toobig, " too big and ",
         toosmall, " too small on a total of ", total, ".\n");
  return (result);
end;

DiamGammaMat := function (size)
```

```
  # Assume env.a is sorted in decreasing order

  local A, M, i;

  A := MatId (Z, size+1);    # One too large, because of KANT bug
  A[1][1] := 2;
  for i in [1..size-1] do
    A[i+1][1] := 1;
    A[i][i+1] := -1;
  od;
  return (MatDecreasedSize (Z, A));  # Restore normal size
end;

DiamGammaVec := function (env, magn)
  # Assume env.a is sorted in decreasing order

  local M, H, G, i;

  M := MatInv (DiamGammaMat (env.n-1));
  G := M * Mat (R, List ( [1..env.n-1], x -> [ env.a[x]^(1/env.k) ] ));
  H := [ ];
  for i in [1..env.n-1] do
    H [i] := [ Round (G[i][1] * magn / env.a[i]^(1/env.k)) ];
  od;
  H [env.n] := [0];
  return (Mat (Z, H));

#   H [i] := [ G[i][1] * magn / env.a[i]^(1/env.k) ];
# od;
# H [env.n] := [0.0];
# return (Mat (R, H));
end;

DiamMagnLarge := function (env, lower, upper)
  local envGamma, ThetaList, Theta, ThetaIndex, i;

  # Code extracted from DiamLarge

  envGamma := Copy (env);
  ThetaList := List ( [1..envGamma.n],
                     x -> envGamma.k / (x+1) *
                          Sum ( envGamma.a{[1..x]}, y -> y^(1/envGamma.k) ));
  Theta := ThetaList[1];
  ThetaIndex := 1;
  for i in [1..Length(ThetaList)] do
    if ThetaList[i] > Theta then
      Theta := ThetaList[i];
      ThetaIndex := i;
    fi;
  od;
  envGamma.a := envGamma.a {[1..ThetaIndex]};
  if envGamma.a [ThetaIndex] <> 1 then
    Add (envGamma.a, 1);
  else
    Add (envGamma.a, 2/3);
  fi;
  envGamma.n := ThetaIndex+1;
```

```
  lower := Ceil ( (lower/envGamma.a[envGamma.n])^(1/envGamma.k) );
  upper := Floor ( (upper/envGamma.a[envGamma.n])^(1/envGamma.k) );
  return (Ceil ( upper * lower^(1/(envGamma.n-1)) ));
end;

DiamLarge := function (env, minratio, lower, upper, magn)
  # Assume env.a is sorted in decreasing order, or at least such that the
  # subset which defines \vartheta is in the first ThetaIndex entries

  local envGamma, ThetaList, Theta, ThetaIndex, i,
        L, G, expo, result, b, i, chi, phi, X, diam, step, dt;

  if (upper+.0)/lower > 10^(env.k-1) then
    Print ("WARNING: bounds differ by more than ", env.k-1, " magnitude(s)\n");
  fi;

  # First determine the 'winning' subset of the a_i, by calculating \vartheta

  envGamma := Copy (env);
  ThetaList := List ( [1..envGamma.n],
                      x -> envGamma.k / (x+1) *
                           Sum ( envGamma.a{[1..x]}, y -> y^(1/envGamma.k) ));
  Theta := ThetaList[1];
  ThetaIndex := 1;
  for i in [1..Length(ThetaList)] do
    if ThetaList[i] > Theta then
      Theta := ThetaList[i];
      ThetaIndex := i;
    fi;
  od;

  # Now we know that m = ThetaIndex and j_{r_s} = s, for s=1, ..., m.
  # We will adapt a new environment from our old environment in such a way that
  # there remain m+1 a_i's, with n = m+1. This is because the dimension of the
  # lattice we will use is equal to m+1 and not necessarily to env.n. Depending
  # on m some a_i's will be discarded, and an extra a_i will be added. Compare
  # the proof of Proposition 2.3.

  envGamma.a := envGamma.a {[1..ThetaIndex]};
  if envGamma.a [ThetaIndex] <> 1 then
    Add (envGamma.a, 1);
  else
    Add (envGamma.a, 2/3);
  fi;
  envGamma.n := ThetaIndex+1;

  # Determine the correct upper and lower bounds that the coefficients of
  # lattice points should satisfy in order to be included in our enumeration.
  # Remember that the coefficient of the m'th coordinate (it makes no
  # difference whether we take it respective to the original base of our
  # DiamLattice, or respective to the LLL-reduced basis, because the bottom
  # row of the defining matrix is the m'th unit vector) gives the corresponding
  # x to be powered k times and multiplied with envGamma.a[envGamma.n], to
  # obtain our X.

  lower := Ceil ( (lower/envGamma.a[envGamma.n])^(1/envGamma.k) );
  upper := Floor ( (upper/envGamma.a[envGamma.n])^(1/envGamma.k) );
```

```
  if magn=0 then
    magn := Ceil ( upper * lower^(1/(envGamma.n-1)) );
    # Remember that we constructed envGamma such that envGamma.n = m+1.
  fi;

  Print ("Now using environment ", envGamma, "\n");
  Print ("Furthermore: lower = ", lower, " upper = ", upper, " magn = ",
         magn, "\n");

  # Now let's construct our lattice and our lattice reference point...

  L := LatLLL (DiamLattice (envGamma, magn));
  G := LatElt (L, DiamGammaVec (envGamma, magn));

  # ... and set up the enumeration process

  LatEnumReset (L);
  LatEnumRefVec (L, G);
  b := LatShortestElt (L)[1];
  LatEnumUpperBound (L, envGamma.n * upper^2);
    # If all components of our vector should be no larger than upper, we may
    # begin enforcing this by restricting the length of our vector in obvious
    # ways, viz. by setting the enumeration upper bound.
  expo := DiamHiExp (env);
  result := [];

  # Run!

  while LatEnum (L) do
    chi := LatEnumElt (L); # Get new lattice point
    Print ("Lattice point ", chi);
    phi := LatEltVec (chi); # Get corresponding R-vector

    Print (MatTrans (phi), "\n");

    if phi[envGamma.n][1] < 0 then phi := -phi; fi;

#    b := Abs (phi[1][1]) < upper;
#    for i in [2..envGamma.n] do
#      b := b and (Abs (phi[i][1]) < upper);
#    od;
    if phi[envGamma.n][1] < upper and lower <= phi[envGamma.n][1] then
        # We may conclude that our lattice point is close enough to the
        # reference vector G

#     Print ((LatEltLength (chi)+.0) / (envGamma.n*upper^2), " ", chi);

      # Define X

      X := Floor (envGamma.a[envGamma.n] * phi[envGamma.n][1]^env.k);
      diam := Diam (X, env);  # Heed! Now we return to the original env

      if diam[1] > minratio * X^expo then  # Example found!
#        if false then
         if Diam (X+1, env)[1] > diam[1] or
            Diam (X-1, env)[1] > diam[1] then  # Maximum lookup needed?
          Print ("Diam before maximum lookup: ", diam[1], "\n");
```

```
          step := diam[2][env.n]^(env.k-1);  # Calculate step size
          i := 0;
          while step > 1 do
            dt := Diam (X+step, env); i := i+1;
            while dt[1] > diam[1] do
              Print ("Adding ", step, ", N.D. ", dt[1], ". ");
              diam := Copy (dt);
              X := X+step;
              dt := Diam (X+step, env);  i := i+1;
            od;
            dt := Diam (X-step, env); i := i+1;
            while dt[1] > diam[1] do
              Print (step, " off, N.D. ", dt[1], ". ");
              diam := Copy (dt);
              X := X-step;
              dt := Diam (X-step, env); i := i+1;
            od;
            step := Floor (step/10);
          od;
          dt := Diam (X+1, env); i := i+1;
          while dt[1] > diam[1] do
            Print ("Adding 1. ");
            diam := Copy (dt);
            X := X+1;
            dt := Diam (X+1, env); i := i+1;
          od;
          dt := Diam (X-1, env); i := i+1;
          while dt[1] > diam[1] do
            Print ("1 off. ");
            diam := Copy (dt);
            X := X-1;
            dt := Diam (X-1, env); i := i+1;
          od;
          Print ("Diam after maximum lookup (", i, " tries): ", diam[1], "\n");
        else
          Print ("Diam ", diam[1], " is optimal.\n");
        fi;

        # Finally add our optimal example to the catalogue

        Add (result, [ LatEltToList (chi-b), X, diam[2], diam[1],
             diam[1]/X^expo ]);
      fi;             # Diameter test
    fi;               # Component size test
  od;                 # Enumeration loop
  return (result);
end;                  # Function finished

DiamDioSystem := function (line, Gamma, env)
  local ThetaList, Theta, ThetaIndex, xx, i;

  ThetaList := List ( [1..env.n],
                      x -> env.k / (x+1) *
                           Sum ( env.a{[1..x]}, y -> y^(1/env.k) ));
  Theta := ThetaList[1];
  ThetaIndex := 1;
  for i in [1..Length(ThetaList)] do
```

```
    if ThetaList[i] > Theta then
      Theta := ThetaList[i];
      ThetaIndex := i;
    fi;
  od;

  xx := [];
  for i in [1..ThetaIndex] do
    xx[i] := env.a_roots[env.n]/env.a_roots[i] * line[2][env.n] -
                   line[2][i] - Gamma[i][1];
#    xx[i] := xx[i]*xx[i];
  od;
  return (xx);
end;
```

Appendix B

Maple sources

B.1 The Chinese Remainder Theorem for polynomials

The following is the MapleV3 code used to implement the algorithmic proof of this Theorem. It is by no means sophisticated, in that it does not implement the improvements suggested by Cohen [9, Section 1.3.3].

```
# Calling sequence:
#   polcrt (b, pol, x)

# Parameters:
#   b   is a list of polynomial evaluations
#   pol is a list of modulus polynomials
#   x   is the common indeterminate of all polynomials in b and pol

# This routine implements the Chinese Remainder Theorem for polynomials
# over arbitrary fields. If polynomials with whole coefficients are
# supplied, the answer might have rational coefficients.

# Described precisely: let l be the vector length of both b and pol.
# Then this function finds a polynomial f of degree less than the degree of
# the product of all moduli, which satisfies the equations

#   f = b[i] (mod pol[i]), i = 1..l

polcrt := proc (b, pol, x)
  local M,      # array, M[i] contains product of all moduli except ith
        y,      # array, y[i] is polynomial inverse of M[i] modulo pol[i]
        l,      # length of all lists and arrays
        i,      # counter
        prod,   # product of all moduli
        result; # outcome, resulting polynomial

  l := linalg[vectdim](b):
  if l <> linalg[vectdim](pol) then
    ERROR(`lists of different lengths`):
  fi:
  if l = 0 then
    ERROR(`too few arguments`):
  fi:
  prod := 1:
```

```
  for i from 1 to l do
    prod := prod * pol[i]
  od:
  y := array (1..l);
  M := array (1..l);
  for i from 1 to l do
    if printlevel > 1 then
      print (`Now doing `.i.`th modular inversion`)
    fi:
    if b[i] <> 0 then
      M [i] := rem (prod / pol [i], pol [i], x):
      gcdex (M [i], pol [i], x, y [i], 's'):     # assume gcd = 1
    else
      M [i] := 0:
      y [i] := 0:
    fi
  od:
  result := 0;
  for i from 1 to l do
    result := result + b [i] * y [i] * (prod / pol [i]):
  od:
  RETURN(result):
end:
```

B.2 Repeating the experiments of M.A. Bennett

This worksheet relates to the paper 'Simultaneous Rational Approximations to Binomial Functions' by M.A. Bennett, published in *Trans. AMS*, Vol. **348**, No. 5, May 1996, and was put together to be able to do concrete computation examples of the Theorems in this paper, notably Theorem 1.1:

Let $a_0, a_1, \ldots, a_m$ be distinct integers with $a_r = 0$ for some r and $N > max_{0 \le u \le m}|a_u|$ an integer. If, further, s and n are relatively prime with $1 \le s < n$, $\epsilon > 0$ and $c_7 \cdot c_8 < c_4 < c_5 \cdot c_8$, then

$$max_{0 \le u \le m, a_u \neq 0} \left\{ \left| \left(1 + \frac{a_u}{N}\right)^{s/n} - \frac{p_u}{q} \right| \right\} > q^{-\lambda - \epsilon}$$

for all integers $p_0, \ldots, p_m$ and with $q \ge q_0(\epsilon, s, n, a_0, \ldots, a_m, N)$, where $\lambda = 1 + \dfrac{\log(c_4/(c_7 \cdot c_8))}{\log(c_5 \cdot c_8/c_4)}$ and q_0 is effectively computable.

When using this worksheet, you should supply the values to all the defining parameters (namely, m, s, n, $a_0, \ldots, a_m$, and N). Then, on executing the worksheet, Maple will calculate the values of Bennett's constants c_1 to c_8, and eventually the value of the simultaneous irrationality measure λ. Several checks are carried out in the process, each of which should yield *true*.

```
> m:=2;
```

$$m := 2$$

```
> s := 1;
```

$$s := 1$$

```
> n := 2;
```

$$n := 2$$

```
> evalb( igcd (s, n) = 1 and 1 <= s and s < n );
```

$$true$$

It is required that $(s, n) = 1$.

```
> a:=array(0..m, [-1,0,1]): a['i']$'i'=0..m;
```

$$-1, 0, 1$$

```
> N := 49;
```

$$N := 49$$

```
> evalb( N > max ( seq ( abs ( a[u] ), u=0..m ) ) );
```

$$true$$

It is required that N be greater than the maximum of the absolute values of the a_u's.

```
> seq ( abs ( product ( a['l'] - a[v], 'l'=0..v-1 ) * product ( a['l'] - a[v], 'l'=v+1..m ) ), v=0..m );
```

$$2, 1, 2$$

```
> c1 := ilcm (");
```

$$c1 := 2$$

```
> seq ( seq ( abs ( a[l] - a[v] ), l=v+1..m ), v=0..m-1) ;
```

$$1, 2, 1$$

```
> c2 := ilcm (");
```

$$c2 := 2$$

```
> fs := numtheory[factorset](n):
> c3 := product ( fs[i]^max ( padic[ordp] (n/c2, fs[i])+1/(fs[i]-1), 0 ),i=1..nops(fs));
```

$$c3 := 2$$

```
> c4 := c1*c2*c3;
```

$$c4 := 8$$

```
> A := z -> product ( z - a['u'], 'u'=0..m );
```

$$A := z \to \prod_{'u'=0}^{m} (z - a_{'u'})$$

```
> expand(A(z));
```

$$z^3 - z$$

```
> Aprime := m -> subs(z=m, diff (A(z),z));
```

$$Aprime := m \to \mathrm{subs}\left(z = m, \frac{\partial}{\partial z}\,\mathrm{A}(\,z\,)\right)$$

```
> expand(Aprime(z));
```

$$3\,z^2 - 1$$

```
> Ater := A(z) - (z+N) * diff(A(z),z);
```

$$Ater := (\,z+1\,)\,z\,(\,z-1\,) - (\,z+49\,)\,(\,z\,(\,z-1\,) + (\,z+1\,)\,(\,z-1\,) + (\,z+1\,)\,z\,)$$

```
> expand(Ater);
```

$$-2\,z^3 - 147\,z^2 + 49$$

```
> Ater_rootlist := [ fsolve (Ater) ];
```

$$Ater\_rootlist := [\,-73.49546429, -.5796403918, .5751046846\,]$$

```
> c5 := abs (Aprime (Ater_rootlist[1]));
```

$$c5 := 16203.74982$$

```
> c6 := proc(v)
>   if (v=0) then
>     RETURN( abs (Aprime (Ater_rootlist[2])));
>   else
>     if v<m then
>       RETURN ( min ( abs (Aprime (Ater_rootlist[v+1])), abs (Aprime (Ater_rootlist[v+2]))));
>     else
>       RETURN ( abs (Aprime (Ater_rootlist[m+1])));     # v=m
```

```
>     fi;
>   fi;
> end:
```

```
> c7 := min ( seq ( abs ( Aprime (Ater_rootlist[u])), u=2..m+1));
```

$$c7 := .0077638052$$

```
> c8sum := 0:
> for r from 1 to n-1 do
>   if igcd(r,n)=1 then
>    c8sum := c8sum + Psi ( max ( (n*m-r)/(n*m), r/n ) );
>   fi;
> od;
> c8 := exp ( -gamma - c8sum / numtheory[phi](n) );
```

$$c8 := \mathrm{e}^{(-\gamma - \Psi(3/4))}$$

```
> evalb(evalf(c7*c8 < c4) and evalf(c4<c5*c8));
```

$$\mathit{true}$$

Both of the above inequalities should hold to validate Bennett's Theorem 1.1. Namely, both numerator and denominator of the fraction occurring in the calculation of λ should be positive.

```
> evalf([c4,c5,c7,c8]);
```

$$[8., 16203.74982, .0077638052, 1.663036611]$$

```
> lambda := 1 + evalf(ln(c4/(c7*c8))/ln(c5*c8/c4));
```

$$\lambda := 1.791543884$$

Now we have (if the above checks were passed):

$$\max_{0 \le u \le m, a_u \neq 0} \left\{ \left| (N + a_u)^{s/n} - \frac{p_u}{q} \right| \right\} > q^{-\lambda - \epsilon},$$

if q is greater or equal to a certain (effectively computable) $q_0 \in \mathbb{Z}$.

Appendix C

C++-sources

C.1 Finding consecutive almost powers

The C++-code below may be used to find all sequences of n consecutive integers X in some interval $[L, R]$ which satisfy

$$\lambda_{n,k}(X) < X^{\frac{(n-1)k}{nk-1}}$$

for some $k \geq 2$. The parameters $n, k, L,$ and R may be given as command line parameters. A fifth parameter Δ is required, indicating once in how many X a diagnostic output line should be printed.

```
// Source: cexamples2.cc
/***************************************************************************
 This C++ program implements the MapleV3 routine below. Let's check who's
 faster!
 CvdW, 5/12/97    --         C++ code starts at line 113; main() at line 163
****************************************************************************/

/* Second version, includes more diagnostic output */

/* Command line arguments: n, k, lower and upper bounds for search, interval
   for giving search diagnostics. Example:

   cexamples2 4 2 1000 100000000 10000

   If the number of parameters is smaller than 5, the user is asked to
   supply the absent ones.

   The output format is Maple-readable. The first line declares a Maple
   variable, e.g. examples4_2, and opens a Maple list for its value. Then one
   line follows for each example found, which takes the form of a list, and
   contains the example's X-value, the relevant k-free parts, and the maximum
   of the latter, which is the same as \lambda_{n,k}(X). All but the last
   such line end with a comma. The last line finally closes the list and ends
   the declaration with a semicolon. Formally, we have:

   first line:       examples<n>_<k> := [
   subsequent lines: [<X>, [<k-free part of X>, ..., <k-free part of X+(n-1)],
                     <maximum of k-free parts>],
                        (no comma on last example line)
   last line:        ];
```

```
*/

// This is a MapleV3 routine, written by Christiaan van de Woestijne, 4/12/97

// Calling sequence:
//   examples(n,k)

// Parameters:
//   n   an integer, n >= 2
//   k   an integer, k >= 2

// The examples function examines all integers i, between the two bounds given
// by the variables exampleslowerbound and examplesupperbound, to find out
// whether i, i+1, ..., i+n-1 form a sequence of integers all having a large
// k'th power divisor. The criterion is:

//             e(n,k)
// w (i+j) < i          for j=0,..,n-1
//  k

// with w (x) the k'th power-free part of an integer x, and e(n,k) given by
//       k

//                (n-1)*k
//    e(n,k) =  ---------
//                 n*k-1

// which is a theoretical upper bound that should be reached by an infinitude
// of 'base' integers i.

// Integers meeting these requirements are printed, together with a list, con-
// taining the k'th power-free parts of i, i+1, ..., i+n-1.
// All operations are performed using only integers, to speed up operations.

/* ****************** (INACTIVE) MapleV3 STARTS HERE **********************

exampleslowerbound := 10;
examplesupperbound := 2000;

examples := proc (n,k)
  local l,i,j,expnum,expdenom,length,expgcd,pow,pfpart:

  // The array l contains a list of k'th power-free parts that seem
  // interesting. If the length of the list reaches n, we know that
  // the n-1 predecessors of our integer i, and i itself, form a
  // sequence that meets the requirements.

  l := array (1..n):      // maintain a list of k'th power-free parts
  expnum := (n-1)*k-1+1:
  expdenom := n*k-1:
  expgcd := igcd(expnum,expdenom):
  if expgcd > 1 then
    expnum := expnum/expgcd:
    expdenom := expdenom/expgcd:
  fi:
  length := 0:
  for i from exampleslowerbound to examplesupperbound do
```

```
    // First part: compute k'th powerfree part of i

    pfpart := i:
    pow := 2^k:
    while irem (pfpart, pow) = 0 do
      pfpart := pfpart/pow:
    od:
    pow := 3^k:
    while irem (pfpart, pow) = 0 do
      pfpart := pfpart/pow:
    od:
    pow := 5^k:
    for j from 5 by 6 while pow <= pfpart do
      while irem (pfpart, pow) = 0 do
        pfpart := pfpart / pow:
      od:
      pow := (j+2)^k:
      if pow <= pfpart then
        while irem (pfpart, pow) = 0 do
          pfpart := pfpart / pow:
        od:
      fi:
      pow := (j+6)^k:
    od:
    l[n] := pfpart:

    // Second part: check upper bound

    if l[n] ^ expdenom < i ^ expnum then  // Interesting case?
      length := length+1:                 // Increase length of interesting list
      if length = n then                  // Full length reached?
        print ([i-n+1, [seq(l[j],j=1..n)]]):     // Output
        length := length-1:               // Forget first entry
      fi:
    else
      length := 0:                        // Alas. Start from scratch!
    fi:
    for j from n-length to n-1 do         // Shift array values to the left
      l[j]:=l[j+1]:
    od:
  od:
  RETURN():
end:

**************** MapleV3 ENDS HERE ***********************************/


/***** C++ STARTS HERE ***********************************************/

#include <iostream.h>
#include <math.h>
#include <stdlib.h>

// Integer gcd function

long gcd (long a, long b) {
```

```
  if (a < 0) a = -a;
  if (b < 0) b = -b;
  while (a && b) {
#ifdef DEBUG
    cout << "In Gcd: (a, b) == (" << a << ", " << b << ")." << endl;
#endif
    if (a > b)
      a %= b;
    else
      b %= a;
  }
  if (a)
    return a;
  else
    return b;
}

// This function puts the fraction num/denom in lowest terms

void expurgate (unsigned int &num, unsigned int &denom) {
  if (denom)
    if (!num)
      denom = 1;
    else {
      unsigned int fgcd = gcd (num, denom);
#ifdef DEBUG
      cout << "In Expurgate: found gcd of " << fgcd << "." << endl;
#endif
      if (fgcd > 1) {
        num /= fgcd;
        denom /= fgcd;
      }
    }
}

// Fast exponentiation of integers

long intpower (long base, long exponent) {
  if (exponent <= 0)
    return 1;
  else {
    long temp = intpower (base, exponent / 2);
    if (exponent % 2)
      return (base * temp * temp);
    else
      return (temp * temp);
  }
}

void main (int argc, char *argv[]) {
  int exampleslowerbound,  // Simulation parameters
      examplesupperbound,
      diag_interval;

  unsigned int n,// Max number of consecutive integers to be found
  k,// Power required of divisor
  i, j,// Counters
```

```
length = 0; // How many cons. ints found?
long powie, pfpart; // Temporaries to speed up calculations
long lambda;          // Evaluate \lambda_{n,k}(i)
double exponent; // Test exponent;

// Provide values for calculation parameters, from command line, if
// present, or from standard input

if (argc > 1)
  n = atoi (argv [1]);
else {
  cerr << "Give #consecutive integers to find: ";
  do {
    cin >> n;
    if (n < 2) cerr << "Please enter amount greater than 1 ";
    } while (n < 2);
  }

if (argc > 2)
  k = atoi (argv [2]);
else {
  cerr << "Give power: ";
  do {
    cin >> k;
    if (k < 2) cerr << "Please enter power greater than 1 ";
    } while (k < 2);
  }

if (argc > 3)
  exampleslowerbound = atoi (argv [3]);
else {
  cerr << "Start search where? (>10): ";
  do {
    cin >> exampleslowerbound;
    if (exampleslowerbound <= 10)
      cerr << "Please enter bound greater than 10 ";
    } while (exampleslowerbound <= 10);
  }

if (argc > 4)
  examplesupperbound = atoi (argv [4]);
else {
  cerr << "End search where? (>" << exampleslowerbound << "): ";
  do {
    cin >> examplesupperbound;
    if (examplesupperbound <= exampleslowerbound)
      cerr << "Please enter bound greater than " << exampleslowerbound << " ";
    } while (examplesupperbound <= exampleslowerbound);
  }

if (argc > 5)
  diag_interval = atoi (argv [5]);
else {
  cerr << "Progress diagnostics per how much tests? (>= 100) ";
  do {
    cin >> diag_interval;
    if (diag_interval < 100) cerr << "Please enter interval greater than 99 ";
```

```
      } while (diag_interval < 100);
    }

  int freepart[n];

    // The array freepart contains a list of k'th power-free parts that seem
    // interesting. If the length of the list reaches n, we know that
    // the n-1 predecessors of our integer i, and i itself, form a
    // sequence that meets the requirements.

#ifdef LARGENUM
// The next six lines apply only in case a large number package is used
  expnum = (n-1)*k-1+1;
  expdenom = n*k-1;
  expurgate (expnum, expdenom); // Divide out gcd, if any

#ifdef DEBUG
  cout << "Exponent: " << expnum << "/" << expdenom << endl;
#endif

#else
// Use standard C++ numerals
  exponent = (double) (n-1)*k / (n*k-1);
#endif

  cout << "examples" << n << '_' << k << " := [" << endl;
    // endl to avoid confusion when progress diagnostics appear before any
    // examples are found

  for (i = exampleslowerbound; i < examplesupperbound; i++) {

    // Provide diagnostic output in Maple-comment form

    if ( i % diag_interval == 0)
      cout << "# Reached " << i << endl;

    // First part; compute k'th powerfree part of i

    pfpart = i;
    powie = 1 << k;                   // 2^k
    while (pfpart % powie == 0) {
      pfpart /= powie;
    }
    powie = intpower (3, k);
    while (pfpart % powie == 0) {
      pfpart /= powie;
    }
    powie = intpower (5, k);
    for (j = 5; powie <= pfpart; j += 6) {
      while (pfpart % powie == 0) {
        pfpart /= powie;
      }
      powie = intpower (j+2, k);
      if (powie <= pfpart) {
        while (pfpart % powie == 0) {
          pfpart /= powie;
        }
```

```
      }
      powie = intpower (j+6, k);
    }
    freepart[n-1] = pfpart;

#ifdef DEBUG
    if (k > 2)
      cout << "I assert that the " << k << "'th power-free part of ";
    else
      cout << "I assert that the squarefree part of ";
    cout << i << " is " << pfpart << "." << endl;
#endif

    // Second part; check upper bound

    if (freepart[n-1] < pow (i, exponent)) {
                                        // Interesting case found?
      length++;                         // Increase length of interesting list
      if (length == 1)                  // Start a new row?
        lambda = pfpart;                // Initialize lambda
      else
        if (pfpart > lambda)            // Check maximality of lambda
          lambda = pfpart;

#ifdef DEBUG
      cout << "That means I found an interesting case, ";
      if (length == 1)
        cout << "one which starts a row." << endl;
      else
        cout << "of which we have already " << length << "." << endl;
#endif

      if (length == n) {                // Full length reached? -> Output
        cout << '[' << i-n+1 << ", [";
        for (j = 0; j < n-1; j++)
          cout << freepart[j] << ", ";
        cout << freepart[n-1] << "], " << lambda << "]," << endl;
        length--;                       // Forget first entry
        if (freepart[0] > lambda) {     // Possibly discard maximum
          lambda = pfpart;
          for (j = 1; j < n-2; j++)     // Calculate valid maximum
            if (lambda < freepart[j])
              lambda = freepart[j];
  } // if : freepart[0] > lambda
        } // if : length == n
      } // if : interesting case
    else
      length = 0;                       // Alas. Start from scratch!
    for (j=n-length; j < n; j++) {      // Shift array values to the left

#ifdef DEBUG
      cout << "I now replace [" << j-1 << "] (" << freepart[j-1]
           << ") by [" << j << "] (" << freepart[j] << ")." << endl;
#endif

      freepart[j-1]=freepart[j];
    } // for
```

```
  } // for i: main loop ended
  cout << '[' << i << ", [";
  for (j=1; j < n; j++)
    cout << i << ", ";
  cout << i << "]]];" << endl; // Make void entry to show search upper bound
}
//************************* C++ ENDS HERE ********************************
```

Appendix D

Proof of Langevin's Théorème 2

We present a complete proof of Théorème 2 of Langevin (cf. [11]), as the original version is rather concise, and appears to be incomplete as well.

Langevin's Théorème 1 states that given any finite subset $E \subset \mathbb{C}$, we can construct a pair (R, S) of polynomials in $\mathbb{C}[X]$ such that

(i) R and S are monic of the same (positive) degree, relatively prime, and not perfect powers;

(ii) we have $r(RS) - 1 = d(R) - d(R - S)$, where $r(f)$ denotes the number of distinct zeroes of a polynomial f, and d denotes its degree;

(iii) we have $E = Z(RS)$, meaning that E equals the zero set of the polynomial RS.

In his second Remarque to the proof of Théorème 1, Langevin gives the basic idea for a construction of the polynomials whose existence is guaranteed by Théorème 1, at least, for subsets of $\mathbb{C}$ consisting only of rationals. Here we give a complete description of this construction, as it appears necessary for finishing the proof of Théorème 2.

Thus let $\{x_1, x_2, \ldots, x_n\} \subset \mathbb{Q}$, $n \geq 2$; the desired pair of polynomials (R, S) is obtained if we can determine positive integers e_i and f_i $(1 \leq i \leq n)$ such that

$$R(X) = \prod_i (X - x_i)^{e_i} = X^d + r_1 X^{d-1} + \ldots + r_d$$

and

$$S(X) = \prod_i (X - x_i)^{f_i} = X^d + s_1 X^{d-1} + \ldots + s_d$$

have their leftmost $n - 1$ coefficients in common, i.e. $r_i = s_i$ for $1 \leq i \leq n - 2$, as both polynomials are monic. This is easily seen (as Langevin points out) to be equivalent to the solution of the linear system

$$\sum_{i=1}^{n} (e_i - f_i) x_i^j = 0, \quad 0 \leq j \leq n - 2,$$

where the first equation guarantees equality of the degrees of R and S: for these degrees, we have $\sum_{i=1}^n e_i = \sum_{i=1}^n f_i = d$.

The matrix $\mathbf{B}$ associated to this system has kernel dimension one, as removing any of its columns yields a Vandermonde matrix of size $n-1$. We can find a basis vector for the kernel by solving the system $\mathbf{A}\vec{y} = \vec{b}$, where $\vec{b}$ is the last column of $\mathbf{B}$, and $\mathbf{A}$ the remaining $(n-1) \times (n-1)$-matrix. Using Cramer's rule, and the fact that $\mathbf{A}$ is a Vandermonde matrix, and remains so if we replace any of its columns by $\vec{b}$, we find

$$y_j = \prod_{\substack{i=1 \\ i \neq j+1}}^{n} \frac{x_n - x_i}{x_{j+1} - x_i} \text{ for } 0 \leq j \leq n - 2. \tag{D.1}$$

It is clear now that $(y_0\ y_1\ \dots\ y_{n-2}\ 1)^T$ generates $\ker \mathbf{B}$; thus $\vec{e}-\vec{f}$ has to be a multiple of it, and such a one that all its coefficients are integral. Furthermore, they should be relatively prime, in order to avoid that R and S be perfect squares. Having thus established $e_i - f_i$ for all i, e_i and f_i follow because R and S should be relatively prime. Thus we may choose $e_i = \max\{e_i - f_i, 0\}$, and $f_i = \min\{0, f_i - e_i\}$; and we have determined R and S up to order.

Two results follow:

Lemma D.1 *If all x_i are integers, then we have $d \leq \frac{n}{2} \cdot \prod_{1\leq i<\ell\leq n} |x_\ell - x_i|$.*

Proof of Lemma D.1. It is clear that multiplying all of $\vec{y}$ by $\prod_{1\leq i<\ell\leq n-1} |x_\ell - x_i|$ yields an integer vector in $\ker \mathbf{B}$, with all entries bounded from above by $\prod_{1\leq i<\ell\leq n} |x_\ell - x_i|$. Furthermore, our degree d is equal to half of the sum of the absolute values of the entries. The result follows. $\square$

Lemma D.2 *If $x_i = i$ for $1 \leq i \leq n$, then we have $e_i - f_i = (-1)^{n-i}\binom{n-1}{i-1}$, and $d = 2^{n-2}$.*

This is Langevin's Corollaire 1 to his Théorème 1.

Proof of Lemma D.2. We have $y_j = (-1)^{n-j-1}\binom{n-1}{j}$ for $0 \leq j \leq n-2$; thus we may take $e_i - f_i = y_{i-1}$, for $1 \leq i \leq n-1$, and $e_n - f_n = 1$. Summing these in absolute value, and dividing by 2, we find

$$d = \frac{1}{2}\sum_{\ell=0}^{n-1} \binom{n-1}{\ell} = 2^{n-2}.$$

$\square$

We now proceed to the actual proof of Théorème 2.

Proof of Lemma 5.11. Apply Théorème 1 to $E = \{-b_1, \dots, -b_n\}$; it follows that $r(RS) - 1 = n - 1 = d(R) - d(R - S)$, or equivalently,

$$\frac{|R(X)|}{|R(X) - S(X)|} \sim X^{n-1} \text{ for any } X \in \mathbb{R}. \tag{D.2}$$

From the *abc*-conjecture, we have that for any positive δ, there exists a positive constant C_δ such that $X \in \mathbb{Z}$,

$$(R(X) - S(X))\, r(R(X)S(X)) \geq C_\delta\, (R(X))^{1-\delta}$$

for all X (we need not bother as to which of $R(X)$ and $S(X)$ is greatest, as R and S have the same degree).

By Lemma D.1 we may bound $R(X)$ from above by X^D for some positive constant D. Thus, taking $0 < \delta < \varepsilon/D$, we have

$$|r(R(X)S(X))| \geq C_\delta \frac{|R(X)|^{1-\delta}}{|R(X) - S(X)|} \geq C_\delta C' \left|\frac{X^{n-1}}{R(X)^\delta}\right| \geq C_\delta C' X^{n-1-D\delta} \geq X^{n-1-\varepsilon},$$

where the positive constant C' follows from the asymptotic equivalence (D.2). $\square$

Bibliography

[1] Tom M. Apostol, *Calculus*, Second Edition, Volume II, Wiley, New York etc., 1969, p. 411–413.

[2] A. Baker, "The Theory of Linear Forms in Logarithms", in: A. Baker and D.W. Masser (eds.), *Transcendence Theory: Advances and Applications*, London-New York-San Francisco, 1977, 1–27.

[3] A. Baker and H. Davenport, "The equations $3x^2 - 2 = y^2$ and $8x^2 - 7 = z^2$", *Quart. J. of Math. Oxford (2)* **20** [1969], 129–137.

[4] Michael A. Bennett, "Simultaneous Approximation to Pairs of Algebraic Numbers", in: *Proceedings of the 4th Conference of the Canadian Number Theory Association*, CMS Conference Proc. Vol. **15** [1995], AMS, Providence, Rhode Island

[5] Michael A. Bennett, "Simultaneous Rational Approximation to Binomial Functions", *Transactions AMS* **348** [1996], 1717–1738.

[6] J. Browkin and J. Brzeziński, "Some remarks on the *abc*-conjecture", *Math. Comp.* **62** [1994], 931–939.

[7] Y. Bugeaud and K. Győry, "Bounds for the solutions of Thue-Mahler equations and norm form equations", *Acta Arith.* **74** [1996], 273–292.

[8] G.V. Chudnovsky, "On the method of Thue-Siegel", *Ann. Math. (2)* **117** [1983], 325–382.

[9] Henri Cohen, *A Course in Computational Algebraic Number Theory*, 2nd ed., Springer-Verlag, Berlin-Heidelberg-New York, 1995.

[10] Henri Darmon and Loïc Merel, "Winding quotients and some variants of Fermat's Last Theorem", *J. reine angew. Math.* **490** [1997], 81–100.

[11] M. Langevin, "Cas d'égalité pour le Théorème de Mason et applications de la conjecture (*abc*)", *C.R. Acad. Sci. Paris, Série 1* **317** [1993], 441–444.

[12] L.J. Mordell, *Diophantine Equations*, Academic Press, London-New York, 1969, p. 207–208.

[13] J.H. Rickert, "Simultaneous rational approximations and related diophantine equations", *Proc. Cambridge Philos. Soc.* **113** [1993], 461–472.

[14] W.M. Schmidt, *Diophantine Approximation*, Lecture Notes in Mathematics Vol. 785, Springer-Verlag, Berlin-Heidelberg-New York, 1980.

[15] M.M. Sweet, "A Theorem in Diophantine Approximations", *J. Num. Th.* **5** [1973], 245–251.

[16] Jan Turk, "Almost powers in short intervals", *Arch. Math.* **43** [1984], 157–166.

[17] N. Tzanakis and B.M.M. de Weger, "On the practical solution of the Thue equation", *J. Num. Th.* **31** [1989], 99–132.

[18] B.M.M. de Weger, *Algorithms for Diophantine Equations*, CWI Tract 65, Centrum voor Wiskunde en Informatica, Amsterdam, 1989, 19–26.