Analyzing Blockwise Lattice Algorithms Using Dynamical Systems

Guillaume Hanrot[1], Xavier Pujol[1], and Damien Stehlé[2]

[1] ÉNS Lyon, Laboratoire LIP (U. Lyon, CNRS, ENS Lyon, INRIA, UCBL), 46 Allée d'Italie, 69364 Lyon Cedex 07, France
[2] CNRS, Laboratoire LIP (U. Lyon, CNRS, ENS Lyon, INRIA, UCBL), 46 Allée d'Italie, 69364 Lyon Cedex 07, France
{guillaume.hanrot,xavier.pujol,damien.stehle}@ens-lyon.fr

Abstract. Strong lattice reduction is the key element for most attacks against lattice-based cryptosystems. Between the strongest but impractical HKZ reduction and the weak but fast LLL reduction, there have been several attempts to find efficient trade-offs. Among them, the BKZ algorithm introduced by Schnorr and Euchner [FCT'91] seems to achieve the best time/quality compromise in practice. However, no reasonable complexity upper bound is known for BKZ, and Gama and Nguyen [Eurocrypt'08] observed experimentally that its practical runtime seems to grow exponentially with the lattice dimension. In this work, we show that BKZ can be terminated long before its completion, while still providing bases of excellent quality. More precisely, we show that if given as inputs a basis $(\boldsymbol{b}_i)_{i\leq n} \in \mathbb{Q}^{n\times n}$ of a lattice L and a block-size β, and if terminated after $\Omega\left(\frac{n^3}{\beta^2}(\log n + \log\log \max_i \|\boldsymbol{b}_i\|)\right)$ calls to a β-dimensional HKZ-reduction (or SVP) subroutine, then BKZ returns a basis whose first vector has norm $\leq 2\nu_\beta^{\frac{n-1}{2(\beta-1)}+\frac{3}{2}} \cdot (\det L)^{\frac{1}{n}}$, where $\nu_\beta \leq \beta$ is the maximum of Hermite's constants in dimensions $\leq \beta$. To obtain this result, we develop a completely new elementary technique based on discrete-time affine dynamical systems, which could lead to the design of improved lattice reduction algorithms.

Keywords: Euclidean lattices, BKZ, lattice-based cryptanalysis.

1 Introduction

A (full-rank) n-dimensional lattice $L \subseteq \mathbb{R}^n$ is the set of integer linear combinations $\sum_{i=1}^n x_i \boldsymbol{b}_i$ of some linearly independent vectors $(\boldsymbol{b}_i)_{i\leq n}$. Such vectors are called a basis and we write $L = L[(\boldsymbol{b}_i)_i]$. Since L is discrete, it contains a shortest non-zero lattice vector, whose norm $\lambda_1(L)$ is called the lattice minimum. Computing such a vector given a basis is referred to as the (computational) Shortest Vector Problem (SVP), and is NP-hard under randomized reductions [1,12]. The complexities of the best known SVP solvers are no less than exponential [22,23,2,15] (the record is held by the algorithm from [22], with complexity $2^{2n+o(n)} \cdot \mathcal{P}oly(\log \max_i \|\boldsymbol{b}_i\|)$). Finding a vector reaching $\lambda_1(L)$

P. Rogaway (Ed.): CRYPTO 2011, LNCS 6841, pp. 447–464, 2011.

is polynomial-time equivalent to computing a basis of L that is reduced in the sense of Hermite-Korkine-Zolotarev (HKZ). The aforementioned SVP solvers can all be used to compute HKZ-reduced bases, in exponential time. On the other hand, bases reduced in the sense of Lenstra-Lenstra-Lovász (LLL) can be computed in polynomial time [16], but the first vector is only guaranteed to satisfy the weaker inequality $\|\boldsymbol{b}_1\| \leq (4/3+\varepsilon)^{\frac{n-1}{2}} \cdot \lambda_1(L)$ (for an arbitrary $\varepsilon > 0$). In 1987, Schnorr introduced time/quality trade-offs between LLL and HKZ [33]. In the present work, we propose the first analysis of the BKZ algorithm [36,37], which is currently the most practical such trade-off [40,8].

Lattice reduction is a popular tool in cryptanalysis [27]. For many applications, such as Coppersmith's method for computing the small roots of polynomials [5], LLL-reduction suffices. However, reductions of much higher quality seem required to break lattice-based cryptosystems. Lattice-based cryptography originated with Ajtai's seminal hash function [1], and the GGH and NTRU encryption schemes [9,14]. Thanks to its excellent asymptotic performance, provable security guarantees, and flexibility, it is currently attracting wide interest and developing at a steady pace. We refer to [21,31] for recent surveys. A major obstacle to the real-life deployment of lattice-based cryptography is the lack of a precise understanding of the limits of the best practical attacks, whose main component is the computation of strongly reduced lattice bases. This prevents from having a precise correspondence between specific security levels and practical parameters. Our work is a step towards a clearer understanding of BKZ, and thus of the best known attacks.

Strong lattice reduction has been studied for about 25 years (see among others [33,37,34,6,32,8,7]). From a theoretical perspective, the best known time/quality trade-off is due to Gama and Nguyen [7]. By building upon the proof of Mordell's inequality on Hermite's constant, they devised the notion of *slide reduction*, and proposed an algorithm computing slide-reduced bases: Given an arbitrary basis $B = (\boldsymbol{b}_i)_{i \leq n}$ of a lattice L, the slide-reduction algorithm finds a basis $(\boldsymbol{c}_i)_{i \leq n}$ of L such that

$$\|\boldsymbol{c}_1\| \leq ((1+\varepsilon)\gamma_\beta)^{\frac{n-\beta}{\beta-1}} \cdot \lambda_1(L), \tag{1}$$

within $\tau_{\texttt{slide}} := O\left(\frac{n^4}{\beta \cdot \varepsilon} \cdot \log \max_i \|\boldsymbol{b}_i\|\right)$ calls[1] to a β-dimensional HKZ-reduction algorithm and a β-dimensional (computational-)SVP solver, where $\gamma_\beta \approx \beta$ is the β-dimensional Hermite constant. If $L \subseteq \mathbb{Q}^n$, the overall cost of the slide-reduction algorithm is $\leq \mathcal{P}oly(n, \mathrm{size}(B)) \cdot \mathcal{C}_{\texttt{HKZ}}(\beta)$, where $\mathcal{C}_{\texttt{HKZ}}(\beta) = 2^{O(\beta)}$ is the cost of HKZ-reducing in dimension β. The higher β, the lower the achieved SVP approximation factor, but the higher the runtime. Slide reduction also provides a constructive variant of Minkowski's inequality, as (letting $\det L$ denote $\mathrm{vol}(\mathbb{R}^n/L)$):

$$\|\boldsymbol{c}_1\| \leq ((1+\varepsilon)\gamma_\beta)^{\frac{n-1}{2(\beta-1)}} \cdot (\det L)^{\frac{1}{n}}, \tag{2}$$

[1] The component $\frac{n^4}{\beta}$ of this upper bound is derived by adapting the results from [7] to our notations. A more thorough analysis leads to a smaller term.

From a practical perspective, however, slide reduction seems to be (significantly) outperformed by the BKZ algorithm [8]. BKZ also relies on a β-dimensional HKZ-reduction algorithm (resp. SVP-solver). The worst-case quality of the bases it returns has been studied in [34] and is comparable to that of the slide reduction algorithm. The first vector of the output basis $(\boldsymbol{c}_i)_{i\leq n}$ satisfies $\|\boldsymbol{c}_1\| \leq ((1+\varepsilon)\gamma_\beta)^{\frac{n-1}{\beta-1}} \cdot \lambda_1(L)$. Note that this bound essentially coincides with (1), except for large values of β. A bound similar to that of (2) also holds.[2] In practice, the quality of the computed bases seems much higher with BKZ than with the slide-reduction algorithm [8]. With respect to run-time, no reasonable bound is known on the number of calls to the β-dimensional HKZ reduction algorithm it needs to make before termination.[3] In practice, this number of calls does not seem to be polynomially bounded [8] and actually becomes huge when $\beta \geq 25$. Because of its large (and somewhat unpredictable) runtime, it is folklore practice to terminate BKZ before the end of its execution, when the solution of the problem for which it is used for is already provided by the current basis [38,24].

Our result. We show that if terminated within polynomially many calls to HKZ/SVP, a slightly modified version of BKZ (see Section 3) returns bases whose first vectors satisfy a slightly weaker variant of (2).

Theorem 1. *There exists*[4] $C > 0$ *such that the following holds for all* n *and* β. *Let* $B = (\boldsymbol{b}_i)_{i\leq n}$ *be a basis of a lattice* L*, given as input to the modified BKZ algorithm of Section 3 with block-size* β*. If terminated after* $\tau_{\mathtt{BKZ}} := C\frac{n^3}{\beta^2}\left(\log n + \log\log \max_i \frac{\|\boldsymbol{b}_i\|}{(\det L)^{1/n}}\right)$ *calls to an HKZ-reduction (or SVP solver) in dimension* β*, the output* $(\boldsymbol{c}_i)_{i\leq n}$ *is a basis of* L *that satisfies (with* $\nu_\beta \leq \beta$ *defined as the maximum of Hermite's constants in dimensions* $\leq \beta$*):*

$$\|\boldsymbol{c}_1\| \leq 2(\nu_\beta)^{\frac{n-1}{2(\beta-1)}+\frac{3}{2}} \cdot (\det L)^{\frac{1}{n}}.$$

If $L \subseteq \mathbb{Q}^n$*, then the overall cost is* $\leq \mathcal{P}oly(n, \mathrm{size}(B)) \cdot \mathcal{C}_{\mathtt{HKZ}}(\beta)$.

By using [18, p. 25], this provides an algorithm with runtime bounded by $\mathcal{P}oly(n, \mathrm{size}(B)) \cdot \mathcal{C}_{\mathtt{HKZ}}(\beta)$ that returns a basis whose first vector satisfies $\|\boldsymbol{c}_1\| \leq 4(\nu_\beta)^{\frac{n-1}{\beta-1}+3} \cdot \lambda_1(L)$, which is only slightly worse than (1). These results indicate that BKZ can be used to achieve essentially the same quality guarantees as slide reduction, within a number of calls to HKZ in dimension β that is no larger than that of slide reduction. Actually, note that $\tau_{\mathtt{BKZ}}$ is significantly smaller than $\tau_{\mathtt{slide}}$, in particular with a dependence with respect to $\max_i \|\boldsymbol{b}_i\|$ that is *exponentially* smaller. It may be possible to obtain a similar bound for the slide-reduction algorithm by adapting our analysis.

[2] In [8], the bound $\|\boldsymbol{c}_1\| \leq (\gamma_\beta)^{\frac{n-1}{2(\beta-1)}+\frac{1}{2}} \cdot (\det L)^{\frac{1}{n}}$ is claimed to hold, but without proof nor reference. We prove a (slightly) weaker bound, but we are able to improve it if γ_n is replaced by any linear function. See the appendix of the full version [10].

[3] A bound $(n\beta)^n$ is mentioned in [8]. For completeness, we give a proof of a similar result in the appendix of the full version [10].

[4] The constant C is used to absorb lower-order terms in n, and could be taken small.

To achieve our result, we use a completely new approach for analyzing lattice reduction algorithms. The classical approach to bound their runtimes was to introduce a quantity, sometimes called potential, involving the current Gram-Schmidt norms $\|\boldsymbol{b}_i^*\|$, which always strictly decreases every time some elementary step is performed. This technique was introduced by Lenstra, Lenstra and Lovász [16] for analyzing their LLL algorithm, and is still used in all complexity analyses of (variants of) LLL we are aware of. It was later adapted to stronger lattice reduction algorithms [33,6,32,7]. We still measure progress with the $\|\boldsymbol{b}_i^*\|$'s, but instead of considering a single scalar combining them all, we look at the *full vector* $(\|\boldsymbol{b}_i^*\|)_i$. More specifically, we observe that each call to HKZ within BKZ has the effect of applying an affine transformation to the vector $(\log \|\boldsymbol{b}_i^*\|)_i$: instead of providing a lower bound to the progress made on a "potential", we are then led to analyze a discrete-time dynamical affine system. Its fixed-points encode information on the output quality of BKZ, whereas its speed of convergence provides an upper bound on the number of times BKZ calls HKZ.

Intuitively, the effect of a call to HKZ on the vector $(\log \|\boldsymbol{b}_i^*\|)_{i \leq n}$ is to essentially replace β consecutive coefficients by their average. We formalize this intuition by making a specific assumption (see Section 4). Under this assumption, the execution of BKZ exactly matches with a dynamical system that we explicit and fully analyze. However, we cannot prove that this assumption is always correct (counter-examples can actually be constructed). To circumvent this difficulty, we instead consider the vector $\boldsymbol{\mu} = (\frac{1}{i}\sum_{j=1}^{i} \log \|\boldsymbol{b}_j^*\|)_{i \leq n}$. This amortization (also used in [11] for analyzing HKZ-reduced bases) allows us to *rigorously* bound the evolution of $\boldsymbol{\mu}$ by the orbit of a vector under another dynamical system. Since this new dynamical system happens to be a modification of the dynamical system used in the idealized model, the analysis performed for the idealized model can be adapted to the rigorous set-up.

This approach is likely to prove useful for analyzing other lattice reduction algorithms. As an illustration of its power, we provide two new results on LLL. First, we show that the SVP approximation factor $\sqrt{4/3}^{n-1}$ can be reached in polynomial time using only Gauss reductions. This is closely related to the question whether the "optimal LLL" (i.e., using LLL parameter $\delta = 1$) terminates in polynomial time [3,17]. Second, we give a LLL-reduction algorithm of bit-complexity $\mathcal{P}oly(n) \cdot \widetilde{O}(\text{size}(B))$. Such a complexity bound was only very recently achieved, with a completely different approach [29]. Note that close-by results on LLL have been concurrently and independently obtained by Schnorr [35].

Practical aspects. Our result is a (possibly pessimistic) worst-case quality bound on BKZ with early termination. In itself, this does not give a precise explanation of the practical behavior of BKZ. In particular, it does not explain why it outperforms slide reduction, but only why it does not behave significantly worse. However, this study illustrates the usefulness of early termination in BKZ: Much progress is done at the beginning of the execution, and quickly the basis quality becomes excellent; the rest of the execution takes much longer, for a significantly less dramatic quality improvement. This behavior is very clear in practice, as illustrated by Figure 1 of Section 2. Since most of the work

performed by BKZ is completed within the first few calls to HKZ, it shows that the BKZ performance extrapolations used to estimate the hardness of cryptographic instances should focus only on the cost of a single call to HKZ and on the achieved basis quality after a few such calls. For instance, it indicates that the strategy (adopted, e.g., in [14,13]) consisting in measuring the full run-time of BKZ might be reconsidered.

Additionally, parts of the analysis might prove useful to better understand BKZ and devise reduction algorithms with improved practical time/quality trade-offs. In particular, the heuristic modelisation of BKZ as a discrete-time affine dynamical system suggests that the block of vectors on which HKZ-reduction is to be applied could be chosen adaptively, so that the system converges faster to its limit. It would not improve the output quality for BKZ, but it is likely to accelerate its convergence. Also, the second phase of BKZ, the one that takes longer but during which little progress is still made, could be understood by introducing some randomness in the model: most of the time, the norm of the first vector found by the HKZ-reduction sub-routine is around its expected value (a constant factor smaller than its worst-case bound), but it is significantly smaller every now and then. If such a model could predict the behavior of BKZ during its second phase, then maybe it would explain why it outperforms slide reduction. It might give indications on the optimal time for stopping BKZ with block-size β before switching to a larger block-size.

Notations. All vectors will be denoted in bold, and matrices in capital letters. If $\boldsymbol{b} \in \mathbb{R}^n$, the notation $\|\boldsymbol{b}\|$ will refer to its Euclidean norm. If $B \in \mathbb{R}^{n \times n}$, we define $\|B\|_2 = \max_{\|\boldsymbol{x}\|=1} \|B \cdot \boldsymbol{x}\|$ and we denote the spectral radius of B by $\rho(B)$. If B is a rational matrix, we define size(B) as the sum of the bit-sizes of the numerators and denominators of its entries. All complexity statements refer to elementary operations on bits. We will use the Landau notations $o(\cdot)$, $O(\cdot)$, $\widetilde{O}(\cdot)$ and $\Omega(\cdot)$. The notations $\log(\cdot)$ and $\ln(\cdot)$ respectively stand for the base 2 and natural logarithms.

2 Reminders

For an introduction to lattice reduction algorithms, we refer to [28].

Successive Minima. Let L be an n-dimensional lattice. Its i-th minimum $\lambda_i(L)$ is defined as the minimal radius r such that $\mathcal{B}(\mathbf{0}, r)$ contains $\geq i$ linearly independent vectors of L.

Hermite's constant. The n-dimensional Hermite constant γ_n is defined as the maximum taken over all lattices L of dimension n of the quantity $\frac{\lambda_1(L)^2}{(\det L)^{2/\dim(L)}}$. Let $\nu_n = \max_{k \leq n} \gamma_k$, an upper bound on γ_n which increases with n. Very few values of ν_n are known, but we have $\nu_n \leq 1 + \frac{n}{4}$ for all n (see [20, Re 2.7.5]).

Gram-Schmidt orthogonalisation. Let $(\boldsymbol{b}_i)_{i \leq n}$ be a lattice basis. Its Gram-Schmidt orthogonalization $(\boldsymbol{b}_i^*)_{i \leq n}$ is defined recursively by $\boldsymbol{b}_i^* = \boldsymbol{b}_i - \sum_{j<i} \mu_{i,j} \boldsymbol{b}_j^*$ with $\mu_{i,j} = (\boldsymbol{b}_i^*, \boldsymbol{b}_j^*)/\|\boldsymbol{b}_j^*\|^2$ for $i > j$. The $\boldsymbol{b}_i^*$'s are mutually orthogonal. For $i \leq j$, we define $\boldsymbol{b}_j^{(i)}$ as the projection of $\boldsymbol{b}_j$ orthogonally to $\mathrm{Span}(\boldsymbol{b}_k)_{k<i}$. Note that if L is an n-dimensional lattice, then $\det L = \prod_{i=1}^n \|\boldsymbol{b}_i^*\|$, for any basis $(\boldsymbol{b}_i)_{i \leq n}$ of L.

A few notions of reduction. Given a basis $(\boldsymbol{b}_i)_{i\leq n}$, we say that it is *size-reduced* if the Gram-Schmidt coefficients $\mu_{i,j}$ satisfy $|\mu_{i,j}| \leq 1/2$ for all $j < i \leq n$. We say that $(\boldsymbol{b}_i)_{i\leq n}$ is δ-*LLL-reduced* for $\delta \leq 1$ if it is size-reduced and the Lovász conditions $\delta\|\boldsymbol{b}_i^*\|^2 \leq \|\boldsymbol{b}_{i+1}^*\|^2 + \mu_{i+1,i}^2\|\boldsymbol{b}_i^*\|^2$ are satisfied for all $i < n$. For any $\delta < 1$, a δ-LLL-reduced basis of a rational lattice L can be computed in polynomial time, given an arbitrary basis of L as input [16]. We say that $(\boldsymbol{b}_i)_{i\leq n}$ is HKZ-reduced if it is size-reduced and for all $i < n$, we have $\|\boldsymbol{b}_i^*\| = \lambda_1(L[(\boldsymbol{b}_j^{(i)})_{i\leq j\leq n}])$. An HKZ-reduced basis of a lattice $L \subseteq \mathbb{Q}^n$ can be computed in time $2^{2n+o(n)} \cdot \mathcal{P}oly(\text{size}(B))$, given an arbitrary basis B of L as input [22]. The following is a direct consequence of the definitions of the HKZ-reduction and Hermite constant.

Lemma 1. *For any HKZ-reduced basis* $(\boldsymbol{b}_i)_{i\leq n}$*, we have:* $\forall i < n$, $\|\boldsymbol{b}_i^*\| \leq \sqrt{\nu_{n-i+1}} \cdot (\prod_{j=i}^{n} \|\boldsymbol{b}_j^*\|)^{\frac{1}{n-i+1}}$.

The BKZ algorithm. We recall the original BKZ algorithm from [37] in Algorithm 1. BKZ was originally proposed as a mean of computing bases that are almost β-*reduced*. β-Reduction was proposed by Schnorr in [33], but without an algorithm for achieving it. The BKZ algorithm proceeds by iterating tours consisting of $n - 1$ calls to a β-dimensional SVP solver called on the lattices $L[(\boldsymbol{b}_i^{(k)})_{k\leq i\leq k+\beta-1}]$. Its execution stops when no change occurs during a tour.

```
Input  : A (LLL-reduced) basis (b_i)_{i≤n}, a blocksize β and a constant δ < 1.
Output : A basis of L[(b_i)_{i≤n}].
repeat
    for k ← 1 to n − 1 do
        Find b such that ‖b^(k)‖ = λ_1(L[(b_i^(k))_{k≤i≤min(k+β−1,n)}]);
        if δ · ‖b_k^*‖ > ‖b‖ then
            LLL-reduce(b_1, ..., b_{k−1}, b, b_k, ..., b_{min(k+β,n)}).
        else
            LLL-reduce(b_1, ..., b_{min(k+β,n)}).
until no change occurs.
```

Algorithm 1. The Schnorr and Euchner BKZ algorithm

3 Terminating BKZ

In this article, we will not analyze the original BKZ algorithm, but we will focus on a slightly modified variant instead, which is given in Algorithm 2. It also performs BKZ tours, and during a tour it makes $n - \beta + 1$ calls to a β-dimensional HKZ-reduction algorithm. It fits more closely to what would be the simplest BKZ-style algorithm, aiming at producing a basis $(\boldsymbol{b}_i)_{i\leq n}$ such that the projected basis $(\boldsymbol{b}_i^{(k)})_{k\leq i\leq k+\beta-1}$ is HKZ-reduced for all $k \leq n - \beta + 1$.

Differences between the two variants of BKZ. The differences between the two algorithms are the following:

- In Algorithm 2, the execution can be terminated at the end of any BKZ tour.
- In the classical BKZ algorithm, the vector $\boldsymbol{b}$ found by the SVP solver is kept only if $\|\boldsymbol{b}^{(k)}\|$ is smaller than $\delta \cdot \|\boldsymbol{b}_k^*\|$. Such a factor $\delta < 1$ does not appear in Algorithm 2. It is unnecessary for our analysis to hold, complicates the algorithm, and leads to output bases of lesser quality.
- For each k within a tour, Algorithm 1 only requires an SVP solver while Algorithm 2 calls an HKZ-reduction algorithm, which is more complex. We use HKZ-reductions for the ease of the analysis. Our analysis would still hold if the loop was done for k from 1 to $n-1$ and if the HKZ-reductions were replaced by calls to any algorithm that returns bases whose first vector reaches the minimum (which can be obtained by calling any SVP solver, putting the output vector in front of the input basis and calling LLL to remove the linear dependency).
- Finally, to insert $\boldsymbol{b}$ in the current basis, Algorithm 1 performs an LLL-reduction. Indeed, applying LLL inside the projected block (i.e., to $\boldsymbol{b}^{(k)}, \boldsymbol{b}_k^{(k)}, \ldots, \boldsymbol{b}_{k+\beta-1}^{(k)}$) would be sufficient to remove the linear dependency while keeping $\boldsymbol{b}^{(k)}$ in first position, but instead it runs LLL from the beginning of the basis until the end of the next block to be considered (i.e., up to index $\min(k+\beta, n)$. This reduction is performed even if the block is already reduced and no vector is inserted. Experimentally, this seems to improve the speed of convergence of the algorithm by a small factor, but it does not seem easy to use our techniques to analyze this effect.

```
Input  : A basis (b_i)_{i≤n} and a blocksize β.
Output : A basis of L[(b_i)_{i≤n}].
repeat
    for k ← 1 to n − β + 1 do
        Modify (b_i)_{k≤i≤k+β−1} so that (b_i^(k))_{k≤i≤k+β−1} is HKZ-reduced;
        Size-reduce(b_1, ..., b_n).
until no change occurs or termination is requested.
```

Algorithm 2. BKZ', the modified BKZ algorithm

On the practical behavior of BKZ. In order to give an insight on the practical behavior of BKZ and BKZ', we give experimental results on the evolution of the quantity $\frac{\|\boldsymbol{b}_1\|}{(\det L)^{1/n}}$ (the so-called Hermite factor) during their executions. The experiment corresponding to Figure 1 is as follows: We generated 64 knapsack-like bases [25] of dimension $n = 108$, with non-trivial entries of bit-length $100n$; Each was LLL-reduced using `fplll` [4] (with parameters $\delta = 0.99$ and $\eta = 0.51$); Then for each we ran NTL's BKZ [40] and an implementation of BKZ' in NTL, with blocksize 24. Figure 1 only shows the beginning of the executions. For both algorithms, the executions of about half the samples consisted in $\simeq 600$ tours, whereas the longest execution stopped after $\simeq 1200$ tours. The average value of $\frac{\|\boldsymbol{b}_1\|}{(\det L)^{1/n}}$ at the end of the executions was $\simeq 1.012$.

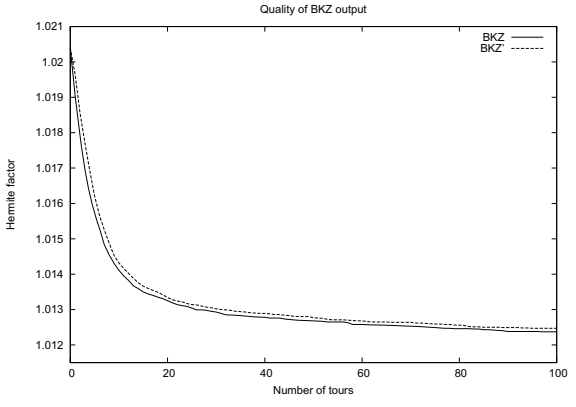

Fig. 1. Evolution of the Hermite factor $\frac{\|\boldsymbol{b}_1\|}{(\det L)^{1/n}}$ during the execution of BKZ and BKZ'

Cost of BKZ'. In order to bound the bit-complexities of BKZ and BKZ', it is classical to consider several cost components separately. In this article, we will focus on the number of tours. The number of calls to an SVP solver (for BKZ) or an HKZ-reduction algorithm (in the case of BKZ') is $\leq n$ times larger. A tour consists of efficient operations (LLL, size-reductions, etc) and of the more costly calls to SVP/BKZ. The cost of the SVP solver or the HKZ-reduction algorithm is often bounded in terms of the number of arithmetic operations it performs: For all known algorithms, this quantity is (at least) exponential in the block-size β. Finally, one should also take into account the bit-costs of the arithmetic operations performed to prepare the calls to SVP/HKZ, during these calls, and after these calls (when applying the computed transforms to the basis, and calling LLL or a size-reduction). These arithmetic costs are classically bounded by considering the bit-sizes of the quantities involved. They can easily be shown to be polynomial in the input bit-size, by relying on rational arithmetic and using standard tools from the analyses of LLL and HKZ [16,15]. It is likely that these costs can be lowered further by relying on floating-point approximations to these rational numbers, using the techniques from [26,30]. To conclude, the overall cost is upper bounded by $\mathcal{P}oly(n, \log \|B\|) \cdot 2^{O(\beta)} \cdot \tau$, where τ is the number of tours.

4 Analysis of BKZ' in the Sandpile Model

In this section, we (rigorously) analyze a *heuristic model of BKZ'*. In the following section, we will show how this analysis can be adapted to allow for a (rigorous) study of the *genuine BKZ'* algorithm.

We first note that BKZ' can be studied by looking at the way the vector $\boldsymbol{x} := (\log \|\boldsymbol{b}_i^*\|)_i$ changes during the execution, rather than considering the whole basis $(\boldsymbol{b}_i)_i$. This simplification is folklore in the analyses of lattice reduction algorithms, and allows for an interpretation in terms of sandpiles [19]. The study in the present section is heuristic in the sense that we assume the effect of a call to HKZ_β on $\boldsymbol{x}$ is determined by $\boldsymbol{x}$ only, in a deterministic fashion.

4.1 The Model and Its Dynamical System Interpretation

Before describing the model, let us consider the shape of a β-dimensional HKZ-reduced basis. Let $(\boldsymbol{b}_i)_{i\leq\beta}$ be an HKZ-reduced basis, and define $x_i = \log \|\boldsymbol{b}_i^*\|$. Then, by Lemma 1, we have:

$$\forall i \leq \beta,\ x_i \leq \frac{1}{2}\log \nu_{\beta-i+1} + \frac{1}{\beta-i+1}\sum_{j=i}^{\beta} x_j. \tag{3}$$

Our heuristic assumption consists in replacing these inequalities by equalities.

Heuristic Sandpile Model Assumption (SMA). We assume for any HKZ-reduced basis $(\boldsymbol{b}_i)_{i\leq\beta}$, we have $x_i = \frac{1}{2}\log \nu_{\beta-i+1} + \frac{1}{\beta-i+1}\sum_{j=i}^{\beta} x_j$ for all $i \leq \beta$, with $\boldsymbol{x} = (\log \|\boldsymbol{b}_i^*\|)_{i\leq\beta}$.

Under SMA, once $\sum_i x_i$ (i.e., $|\det(\boldsymbol{b}_i)_i|$) is fixed, an $\boldsymbol{x}$ of an HKZ-reduced basis is uniquely determined.

Lemma 2. *Let $(\boldsymbol{b}_i)_{i\leq\beta}$ be HKZ-reduced, $\boldsymbol{x} = (\log \|\boldsymbol{b}_i^*\|)_i$ and $\mathbb{E}[\boldsymbol{x}] = \sum_{i\leq\beta}\frac{x_i}{\beta}$. Then, under SMA, $x_\beta = \mathbb{E}[\boldsymbol{x}] - \Gamma_\beta(\beta-1)$ and:*

$$\forall i < \beta,\ x_i = \mathbb{E}[\boldsymbol{x}] - (\beta-i+1)\Gamma_\beta(i-1) + (\beta-i)\Gamma_\beta(i),$$

with $\Gamma_n(k) = \sum_{i=n-k}^{n-1}\frac{\log \nu_{i+1}}{2i}$ for all $0 \leq k < n$.

We now exploit SMA to interpret BKZ' as a discrete-time linear dynamical system. Let $(\boldsymbol{b}_i)_{i\leq n}$ be a lattice basis and $\boldsymbol{x} = (\log \|\boldsymbol{b}_i^*\|)_i$. Let $\beta \leq n$ be a block-size and $\alpha \leq n-\beta+1$. When we apply an HKZ reduction algorithm to the projected sublattice $(\boldsymbol{b}_i^{(\alpha)})_{\alpha\leq i<\alpha+\beta-1}$, we obtain a new basis $(\boldsymbol{b}'_i)_{i\leq n}$ such that (with $\boldsymbol{x}' = (\log \|\boldsymbol{b}_i'^*\|)_i$):

$$\sum_{i=\alpha}^{\alpha+\beta-1} x'_i = \sum_{i=\alpha}^{\alpha+\beta-1} x_i \quad \text{and} \quad \forall i \notin [\alpha, \alpha+\beta-1],\ x'_i = x_i.$$

Under SMA, we also have:

$$\forall i \in [\alpha, \alpha+\beta-1],\ x'_i = \frac{1}{2}\log \nu_{\alpha+\beta-i} + \frac{1}{\alpha+\beta-i}\sum_{j=i}^{\alpha+\beta-1} x'_j.$$

By applying Lemma 2, we obtain $\boldsymbol{x}' = A^{(\alpha)} \cdot \boldsymbol{x} + \boldsymbol{g}^{(\alpha)}$, with:

$$A^{(\alpha)} = \begin{pmatrix} \ddots & & & & & \\ & 1 & & & & \\ & & \frac{1}{\beta} \cdots \frac{1}{\beta} & & & (\alpha) \\ & & \vdots \ddots \vdots & & & \\ & & \frac{1}{\beta} \cdots \frac{1}{\beta} & & & (\alpha+\beta-1) \\ & & & 1 & & \\ & & & & \ddots & \end{pmatrix}$$

$$g_i^{(\alpha)} = \begin{cases} 0 \text{ if } i < \alpha \\ (\beta+\alpha-i-1)\Gamma_\beta(i-\alpha+1) - (\beta+\alpha-i)\Gamma_\beta(i-\alpha) \\ \qquad\qquad\qquad\qquad\qquad\qquad \text{if } i \in [\alpha, \alpha+\beta-2] \\ -\Gamma_\beta(\beta-1) \text{ if } i = \alpha+\beta-1 \\ 0 \text{ if } i \geq \alpha+\beta. \end{cases}$$

We recall that a *BKZ' tour* is the successive $(n-\beta+1)$ applications of an HKZ-reduction algorithm with $\alpha = 1, \ldots, n-\beta+1$ (in this order). Under SMA, the effect of a BKZ' tour on $\boldsymbol{x}$ is to replace it by $A\boldsymbol{x} + \boldsymbol{g}$ with $\boldsymbol{g} = \boldsymbol{g}^{(n-\beta+1)} + A^{(n-\beta+1)} \cdot (\boldsymbol{g}^{(n-\beta)} + A^{(n-\beta)} \cdot (\ldots))$ and:

$$A = A^{(n-\beta+1)} \cdot \ldots \cdot A^{(1)} = \begin{pmatrix} & \overset{(1)}{} & & \overset{(\beta)}{} & & \\ & \frac{1}{\beta} & \cdots & \frac{1}{\beta} & & \\ & \frac{\beta-1}{\beta^2} & \cdots & \frac{\beta-1}{\beta^2} & \frac{1}{\beta} & \\ & \vdots & & \vdots & \ddots & \ddots \\ & \frac{(\beta-1)^{n-\beta}}{\beta^{n-\beta+1}} & \cdots & \frac{(\beta-1)^{n-\beta}}{\beta^{n-\beta+1}} & \cdots \frac{\beta-1}{\beta^2} & \frac{1}{\beta} \\ & \vdots & & \vdots & \vdots & \vdots \\ & \frac{(\beta-1)^{n-\beta}}{\beta^{n-\beta+1}} & \cdots & \frac{(\beta-1)^{n-\beta}}{\beta^{n-\beta+1}} & \cdots \frac{\beta-1}{\beta^2} & \frac{1}{\beta} \end{pmatrix} \begin{matrix} \\ \\ \\ \\ (n-\beta+1) \\ \\ (n) \end{matrix}.$$

We sum up the study of the discrete-time dynamical system $\boldsymbol{x} \leftarrow A \cdot \boldsymbol{x} + \boldsymbol{g}$ in the following Theorem. The solutions and speed of convergence respectively provide information on the output quality and runtime of BKZ' (under SMA). Overall, we have:

Theorem 2. *Under SMA, there exists $C > 0$ such that the following holds for all n and β. Let $(\boldsymbol{b}_i)_{i \leq n}$ be given as input to BKZ'$_\beta$ and L the lattice spanned by the $\boldsymbol{b}_i$'s. If terminated after $C\frac{n^2}{\beta^2}(\log n + \log\log \max_i \frac{\|\boldsymbol{b}_i^*\|}{(\det L)^{1/n}})$ tours, then the output $(\boldsymbol{c}_i)_{i \leq n}$ is a basis of L that satisfies $\|\boldsymbol{x} - \boldsymbol{x}^\infty\|_2 \leq 1$, where*

$x_i = \log \frac{\|\boldsymbol{c}_i^*\|}{(\det L)^{1/n}}$ *for all* i *and* $\boldsymbol{x}^\infty$ *is the unique solution of the equation* $\boldsymbol{x}^\infty = A \cdot \boldsymbol{x}^\infty + \boldsymbol{g}$ *with* $\mathbb{E}[\boldsymbol{x}^\infty] = 0$. *This implies that:*[5]

$$\|\boldsymbol{c}_1\| \leq 2(\nu_\beta)^{\frac{n-1}{2(\beta-1)}+\frac{3}{2}} \cdot (\det L)^{\frac{1}{n}}.$$

4.2 Solutions of the Dynamical System

Before studying the solutions of $\boldsymbol{x} = A \cdot \boldsymbol{x} + \boldsymbol{g}$, we consider the associated homogeneous system.

Lemma 3. *If* $A \cdot \boldsymbol{x} = \boldsymbol{x}$, *then* $\boldsymbol{x} \in \mathrm{span}(1, \ldots, 1)^T$.

It thus suffices to find one solution to $\boldsymbol{x} = A \cdot \boldsymbol{x} + \boldsymbol{g}$ to obtain all the solutions. We define $\overline{\boldsymbol{x}}$ as follows:

$$\overline{x}_i = \begin{cases} \frac{\beta}{2(\beta-1)} \log \nu_\beta + \frac{1}{\beta-1} \sum_{j=i+1}^{i+\beta-1} \overline{x}_j & \text{if } i \leq n-\beta \\ g_i^{(n-\beta+1)} & \text{if } i > n-\beta \end{cases}.$$

Lemma 4. *We have* $\overline{\boldsymbol{x}} = A \cdot \overline{\boldsymbol{x}} + \boldsymbol{g}$.

We now provide explicit lower and upper bounds for the coordinates of the solution $\overline{\boldsymbol{x}}$.

Lemma 5. *For all* $i \leq n-\beta+1$, *we have* $\left(\frac{n-i}{\beta-1} - \frac{3}{2}\right) \log \nu_\beta \leq \overline{x}_i - \overline{x}_{n-\beta+1} \leq \frac{n-i}{\beta-1} \log \nu_\beta$.

We refer to [10] for proofs Lemmata 3, 4, 5.

As the set of solutions to $\boldsymbol{x} = A \cdot \boldsymbol{x} + \boldsymbol{g}$ is $\overline{\boldsymbol{x}} + \mathrm{Span}(1, \ldots, 1)^T$, the value of $\overline{\boldsymbol{x}}$ is only interesting up to a constant vector, which is why we bound $\overline{x}_i - \overline{x}_{n-\beta+1}$ rather than $\overline{x}_i$. In other words, since $\boldsymbol{x}^\infty$ of Theorem 1 is $\overline{\boldsymbol{x}} - (\mathbb{E}[\overline{\boldsymbol{x}}])_i$, the Lemma also applies to $\boldsymbol{x}^\infty$. It is also worth noting that the difference between the upper and lower bounds $\frac{3}{2} \log \nu_\beta$ is much smaller than the upper bound $\frac{n-i}{\beta-1} \log \nu_\beta$ (for most values of i). If we replace ν_β by β, then, via a tedious function analysis, we can improve both bounds so that their difference is lowered to $\frac{1}{2} \log \beta$. In the special case $\beta = 2$, the expression of $\overline{\boldsymbol{x}}$ is $\overline{x}_i = \overline{x}_n + (n-i) \log \nu_2$.

4.3 Speed of Convergence of the Dynamical System

The classical approach to study the speed of convergence (with respect to k) of a discrete-time dynamical system $\boldsymbol{x}_{k+1} := A_n \cdot \boldsymbol{x}_k + \boldsymbol{g}_n$ (where A_n and $\boldsymbol{g}_n$ are the n-dimensional values of A and $\boldsymbol{g}$ respectively) consists in providing an upper bound to the largest eigenvalue of $A_n^T A_n$. It is relatively easy to prove that it is 1 (note that A_n is doubly stochastic). We are to show that the second largest

[5] If we replace ν_β by a linear function that bounds it (e.g., $\nu_\beta \leq \beta$), then the constant $\frac{3}{2}$ may be replaced by $\frac{1-\ln 2}{2} + \varepsilon$ (with $\varepsilon > 0$ arbitrarily close to 0 and β sufficiently large).

singular value is $< 1 - \frac{\beta^2}{2n^2}$, and that this bound is sharp, up to changing the constant $1/2$ and as long as $n - \beta = \Omega(n)$.

The asymptotic speed of convergence of the sequence $(A_n^k \cdot \boldsymbol{x})_k$ is in fact determined by the eigenvalue(s) of A_n of largest module[6] (this is the principle of the power iteration algorithm). However, this classical fact provides no indication on the dependency with respect to $\boldsymbol{x}$, which is crucial in the present situation. As we use the bound $\|A_n^k \cdot \boldsymbol{x}\| \leq \|A_n\|_2^k \cdot \|\boldsymbol{x}\|$, we are led to studying the largest singular values of $A_n^T A_n$.

We first explicit the characteristic polynomial χ_n of $A_n^T A_n$. The following lemma shows that it satisfies a second order recurrence formula.

Lemma 6. *We have* $\chi_\beta(t) = t^{\beta-1}(t-1)$, $\chi_{\beta+1}(t) = t^{\beta-1}(t-1)(t - \frac{1}{\beta^2})$ *and, for any* $n \geq \beta$*:*

$$\chi_{n+2}(t) = \frac{(2\beta(\beta-1)+1)t - 1}{\beta^2} \cdot \chi_{n+1}(t) - \left(\frac{\beta-1}{\beta}\right)^2 t^2 \cdot \chi_n(t).$$

This is used to study the roots of $\chi_n(t)$. The proof of the following result relies on several changes of variables to link the polynomials $\chi_n(t)$ to the Chebyshev polynomials of the second kind.

Lemma 7. *For any* $n \geq \beta \geq 2$*, the largest root of the polynomial* $\frac{\chi_n(t)}{t-1}$ *belongs to* $\left[1 - \frac{\pi^2\beta^2}{(n-\beta)^2},\ 1 - \frac{\beta^2}{2n^2}\right]$.

We refer to [10] for proofs of Lemmata 6 and 7.

Proof of Theorem 2. The unicity and existence of $\boldsymbol{x}^\infty$ come from Lemmata 3 and 4.

Let $(\boldsymbol{b}_i^{(k)})_{i \leq n}$ be the basis after k tours of the algorithm BKZ'$_\beta$ and $\boldsymbol{x}_i^{(k)} = \log \frac{\|\boldsymbol{b}_i^{(k)*}\|}{(\det L)^{1/n}}$. The definition of $\boldsymbol{x}^\infty$ and a simple induction imply that $\boldsymbol{x}^{(k)} - \boldsymbol{x}^\infty = A^k(\boldsymbol{x}^{(0)} - \boldsymbol{x}^\infty)$. Both $\boldsymbol{x}^{(0)}$ and $\boldsymbol{x}^\infty$ live in the subspace $\mathcal{E} := \mathrm{Span}(1, \ldots, 1)^\perp$, which is stabilized by A. Let us denote by $A_\mathcal{E}$ the restriction of A to this subspace. Then the largest eigenvalue of $A_\mathcal{E}^T A_\mathcal{E}$ is bounded in Lemma 7 by $\left(1 - \frac{\beta^2}{2n^2}\right)$. Taking the norm in the previous equation gives:

$$\begin{aligned}\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^\infty\|_2 &\leq \|A_\mathcal{E}\|_2^k \cdot \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\infty\|_2 = \rho(A_\mathcal{E}^T A_\mathcal{E})^{k/2} \cdot \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\infty\|_2 \\ &\leq \left(1 - \frac{\beta^2}{2n^2}\right)^{k/2} \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\infty\|_2.\end{aligned}$$

The term $\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\infty\|_2$ is bounded by $\left(\log \frac{\max_i \|\boldsymbol{b}_i^*\|}{(\det L)^{1/n}}\right) n + n^{O(1)}$. Thus, there exists C such that $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^\infty\|_2 \leq 1$ when $k \geq C\frac{n^2}{\beta^2}(\log n + \log\log \max_i \frac{\|\boldsymbol{b}_i^*\|}{(\det L)^{1/n}})$.

[6] Which can also be proved to be $\leq 1 - c\beta^2/n^2$ for some constant c.

We now prove the last inequality of the theorem. By Lemma 5 and the fact that $\sum_{i=n-\beta+1}^{n} x_i^{\infty} \geq \beta x_{n-\beta+1}^{\infty} + \sum_{i=n-\beta+1}^{n} \left(\frac{\log \nu_\beta}{\beta-1}(n-i) - \frac{3}{2}\log \nu_\beta\right)$, we have:

$$\begin{aligned} x_1^{\infty} &\leq (n-1)\frac{\log \nu_\beta}{\beta-1} - \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\log \nu_\beta}{\beta-1}(n-i) - \frac{3}{2}\log \nu_\beta\right) \\ &= \left(\frac{n-1}{2(\beta-1)} + \frac{3}{2}\right)\log \nu_\beta. \end{aligned}$$

Using the inequality $x_1^{(k)} \leq x_1^{\infty} + 1$ and taking the exponential (in base 2) leads to the result. □

5 Analysis of BKZ'

We now show how the heuristic analysis of the previous section can be made rigorous. The main difficulty stems from the lack of control on the $\|\boldsymbol{b}_i^*\|$'s of an HKZ-reduced basis $(\boldsymbol{b}_i)_{i\leq\beta}$. More precisely, once the determinant and $\|\boldsymbol{b}_\beta^*\|$ are fixed, the $\|\boldsymbol{b}_i^*\|$'s are all below a specific curve (explicitly given in Lemma 2). However, if only the determinant is fixed, the pattern of the $\|\boldsymbol{b}_i^*\|$'s can vary significantly: as an example, taking orthogonal vectors of increasing norms shows that $\|\boldsymbol{b}_1^*\|$ (resp. $\|\boldsymbol{b}_\beta^*\|$) can be arbitrarily small (resp. large). Unfortunately, when applying HKZ within BKZ', it seems we only control the determinant of the HKZ-reduced basis of the considered block, although we would prefer to have an upper bound for each Gram-Schmidt norm individually. We circumvent this difficulty by *amortizing* the analysis over the $\|\boldsymbol{b}_i^*\|$'s: as observed in [11], we have a sharp control *on each average* of the first $\|\boldsymbol{b}_i^*\|$'s. For an arbitrary basis $B := (\boldsymbol{b}_i)_{i\leq n}$, we define $\mu_k^{(B)} = \frac{1}{k}\sum_{1\leq i\leq k}\log\|\boldsymbol{b}_i^*\|$, for $k \leq n$.

Lemma 8 ([11, Le. 3]). *If $B = (\boldsymbol{b}_i)_{i\leq\beta}$ is HKZ-reduced, then $\mu_k^{(B)} \leq \frac{\beta-k}{k}\log \Gamma_\beta(k) + \mu_\beta^{(B)}$ for all $k \leq \beta$.*

5.1 A Dynamical System for (Genuine) BKZ' Tours

We now reformulate the results of the previous section with the $\mu_i^{(B)}$'s instead of the $\log\|\boldsymbol{b}_i^*\|$'s. This amounts to a base change in the discrete-time dynamical system of Subsection 4.1. We define:

$$P = (\tfrac{1}{i}\mathbf{1}_{i\geq j})_{1\leq i,j\leq n},\ \widetilde{A} = PAP^{-1} \text{ and } \widetilde{\boldsymbol{g}} = P\cdot\boldsymbol{g}.$$

Note that $\boldsymbol{\mu}^{(B)} = P\cdot\boldsymbol{x}^{(B)}$, where $\boldsymbol{x}^{(B)} = (\log\|\boldsymbol{b}_i^*\|)_i$ and $\boldsymbol{\mu}^{(B)} = (\mu_i^{(B)})_i$.

Lemma 9. *Let B' be the basis obtained after a BKZ' tour given an n-dimensional basis B as input. Then $\boldsymbol{\mu}^{(B')} \leq \widetilde{A}\cdot\boldsymbol{\mu}^{(B)} + \widetilde{\boldsymbol{g}}$, where the inequality holds componentwise.*

5.2 Analysis of the Updated Dynamical System

Similarly to the analysis of the previous section, it may be possible to obtain information on the speed of convergence of BKZ' by estimating the eigenvalues of $\widetilde{A}^T \cdot \widetilde{A}$. However, the latter eigenvalues seem significantly less amenable to study than those of $A^T A$. The following lemma shows that we can short-circuit the study of the modified dynamical system. For a basis $B \in \mathbb{R}^{n \times n}$ given as input to BKZ'$_\beta$, we define $B^{[0]} = B$ and $B^{[i]}$ as the current basis after the i-th BKZ' tour. We also define $\boldsymbol{\mu}^\infty = P \cdot \boldsymbol{x}^\infty$.

Lemma 10. *Let $B \in \mathbb{R}^{n \times n}$ a basis given as input to BKZ'$_\beta$. Wlog we assume that $\mu_n^{(B)} = \mu_n^\infty$ (since $\mu_n^{(B)} = \frac{1}{n} \log |\det B|$, this can be achieved by multiplying B by a scalar). We have:*

$$\forall k \geq 0, \forall i \leq n, \quad \mu_i^{(B^{[k]})} \leq \mu_i^\infty + (1 + \log n)^{1/2} \cdot \left(1 - \frac{\beta^2}{2n^2}\right)^{k/2} \|\boldsymbol{x}^{(B^{[0]})} - \boldsymbol{x}^\infty\|_2.$$

Lemma 11. *There exists $C > 0$ such that the following holds for all integers $n \geq \beta$, and $\varepsilon \in (0, 1]$. Let $(\boldsymbol{b}_i)_{i \leq n}$ be a basis of a lattice L, given as input to the modified BKZ' algorithm of Section 2 with block-size β. If terminated after $C\frac{n^3}{\beta^2}(\log \frac{n}{\varepsilon} + \log\log \max_i \frac{\|\boldsymbol{b}_i^*\|}{(\det L)^{1/n}})$ calls to an HKZ-reduction (resp. SVP solver) in dimension β, the output $(\boldsymbol{c}_i)_{i \leq n}$ is a basis of L that satisfies:*

$$\|\boldsymbol{c}_1\| \leq (1 + \varepsilon) \nu_\beta^{\frac{n-1}{2(\beta-1)} + \frac{3}{2}} \cdot (\det L)^{\frac{1}{n}}.$$

Theorem 1 corresponds to taking $\varepsilon = 1$ in Lemma 11. Also, when $\beta = 2$, using the explicit expression of $\boldsymbol{x}^\infty$ leads to the improved bound $\|\boldsymbol{c}_1\| \leq (1 + \varepsilon) \cdot (\nu_2)^{\frac{n-1}{2}} \cdot (\det L)^{\frac{1}{n}}$.

6 Applications to LLL-Reduction

In this section, we investigate the relationship between BKZ'$_2$ reduction and the notion of LLL-reduction [16]. Note that analogues of some of the results of this section have been concurrently and independently obtained by Schnorr [35].

Reminders on the LLL algorithm. The LLL algorithm with parameter δ proceeds by successive loop iterations. Each iteration has a corresponding index k, defined as the smallest such that $(\boldsymbol{b}_i)_{i \leq k}$ is not δ-LLL-reduced. The iteration consists in size-reducing $(\boldsymbol{b}_i)_{i \leq k}$ and then checking Lovász's condition $\delta \|\boldsymbol{b}_{k-1}^*\|^2 \leq \|\boldsymbol{b}_k^*\|^2 + \mu_{k,k-1}^2 \|\boldsymbol{b}_{k-1}^*\|^2$. If it is satisfied, then we proceed to the next loop iteration, and otherwise, we swap the vectors $\boldsymbol{b}_k$ and $\boldsymbol{b}_{k-1}$. Any such swap decreases the quantity $\Pi((\boldsymbol{b}_i)_i) = \prod_{i=1}^n \|\boldsymbol{b}_i^*\|^{2(n-i+1)}$ by a factor $\geq 1/\delta$ whereas it remains unchanged during size-reductions. Since $\Pi((\boldsymbol{b}_i)_i) \leq 2^{O(n^2 \operatorname{size}(B))}$ and since for any integer basis $\Pi((\boldsymbol{b}_i)_i)$ is an integer, this allows to prove termination within $O(n^2 \operatorname{size}(B))$ loop iterations when $\delta < 1$. When $\delta = 1$, we obtain the so-called *optimal LLL algorithm*. Termination can still be proven by using different arguments, but with a much larger bound $2^{\mathcal{P}oly(n)} \cdot \mathcal{P}oly(\operatorname{size}(B))$ (see [3,17]).

An iterated version of BKZ'$_2$. We consider the algorithm Iterated-BKZ'$_2$ (described in Algorithm 3) which given as input a basis $(\boldsymbol{b}_i)_{i\leq n}$ successively applies BKZ'$_2$ to the projected bases $(\boldsymbol{b}_i)_{i\leq n}$, $(\boldsymbol{b}_i^{(2)})_{2\leq i\leq n}$, $\ldots$, $(\boldsymbol{b}_i^{(n-1)})_{n-1\leq i\leq n}$. By using a quasi-linear time Gauss reduction algorithm (see [39,41]) as the HKZ_2 algorithm within BKZ'$_2$, Algorithm Iterated-BKZ'$_2$ can be shown to run in quasi-linear time.

```
Input  : A basis (b_i)_{i≤n} of a lattice L.
Output : A basis of L.
for k := 1 to n − 1 do
    Apply BKZ'_2 to the basis (b_i^(k))_{k≤i≤n};
    Let T be the corresponding transformation matrix;
    Update (b_i)_{i≤n} by applying T to (b_i)_{k≤i≤n}.
Return (b_i)_{i≤n}.
```

Algorithm 3. Iterated-BKZ'$_2$ Algorithm

Lemma 12. *Let B be a basis of an n-dimensional lattice, and $\varepsilon > 0$ be arbitrary. Then, using Algorithm Iterated-BKZ'$_2$, one can compute, in time $\mathcal{P}oly(n)\cdot\widetilde{O}(\mathrm{size}(B))$, a basis $(\boldsymbol{b}'_i)_{i\leq n}$ such that*

$$\forall i\leq n,\ \|\boldsymbol{b}_i'^*\|\leq(1+\varepsilon)\left(\frac{4}{3}\right)^{\frac{n-i}{2}}\cdot\left(\prod_{j=i}^{n}\|\boldsymbol{b}_i'^*\|\right)^{\frac{1}{n-i+1}}. \tag{4}$$

A close analogue of the optimal LLL. Let $B=(\boldsymbol{b}_i)_{i\leq n}$ an integral basis output by Iterated-BKZ'$_2$. For $i\leq n$, we let p_i,q_i be coprime rational integers such that $\frac{p_i}{q_i}=\left(\frac{3}{4}\right)^{(n-i+1)(n-i)}\cdot\frac{\|\boldsymbol{b}_i^*\|^{2(n-i+1)}}{\prod_{j=i}^{n}\|\boldsymbol{b}_j^*\|^2}$. By (4), we know that $p_i/q_i\leq(1+\varepsilon)^{n-i+1}$. Note that p_i/q_i is a rational number with denominator $\leq 2^{O(n^2+\mathrm{size}(B))}$. We can thus find a constant c such that, for all i, the quantity $|p_i/q_i-1|$ is either 0 or $\geq 2^{-c(n^2+\mathrm{size}(B))}$. Hence, if we choose $\varepsilon<\frac{1}{2n}.2^{-c(n^2+\mathrm{size}(B'))}$, all the inequalities from (4) must hold with $\varepsilon=0$. Overall, we obtain, in polynomial time and using only swaps and size-reductions, a basis for which (4) holds with $\varepsilon=0$.

A quasi-linear time LLL-reduction algorithm. BKZ'$_2$ can be used to obtain a variant of LLL which given as input an integer basis $(\boldsymbol{b}_i)_{i\leq n}$ and $\delta<1$ returns a δ-LLL-reduced basis of $L[(\boldsymbol{b}_i)_{i\leq n}]$ in time $\mathcal{P}oly(n)\cdot\widetilde{O}(\mathrm{size}(B))$. First, we apply the modification from [18, p. 25] to a terminated BKZ'$_2$ so that the modified algorithm, when given as input an integer basis $(\boldsymbol{b}_i)_{i\leq n}$ and $\varepsilon>0$, returns in time $\mathcal{P}oly(n)\cdot\widetilde{O}(\mathrm{size}(B))$ a basis $(\boldsymbol{b}'_i)_{i\leq n}$ of $L[(\boldsymbol{b}_i)_{i\leq n}]$ such that $\|\boldsymbol{b}'_1\|\leq(1+\varepsilon)^2(4/3)^{n-1}\lambda_1(L)$. The complexity bound holds because the transformation from [18, p. 25] applies BKZ'$_2$ n times on bases whose bit-sizes are $\mathcal{P}oly(n)\cdot\widetilde{O}(\mathrm{size}(B))$.

We iterate this algorithm n times on the projected lattices $(\boldsymbol{b}_i^{(k)})_{k\leq i\leq n}$ so that the output basis $(\boldsymbol{c}_i)_{i\leq n}$ of $L[(\boldsymbol{b}_i)_{i\leq n}]$ satisfies:

$$\forall i\leq n,\ \|\boldsymbol{c}_i^*\|\leq(1+\varepsilon)^2(4/3)^{n-i}\lambda_1(L[(\boldsymbol{b}_j^{(i)})_{i\leq j\leq n}]). \tag{5}$$

It follows from inequalities and the size-reducedness of $(\boldsymbol{c}_i)_{1\leq i\leq n}$ that $\text{size}(C) = \mathcal{P}oly(n)\cdot\text{size}(B)$.

We call δ-LLL' the successive application of the above algorithm based on BKZ'$_2$ and LLL with parameter δ. We are to prove that the number of loop iterations performed by δ-LLL is $\mathcal{P}oly(n)$.

Theorem 3. *Given as inputs a basis $B \in \mathbb{Z}^{n\times n}$ of a lattice L and $\delta < 1$, algorithm δ-LLL' algorithm outputs a δ-LLL-reduced basis of L within $\mathcal{P}oly(n)\cdot\widetilde{O}(\text{size}(B))$ bit operations.*

Proof. With the same notations as above, it suffices to prove that given as input $(\boldsymbol{c}_i)_{i\leq n}$, algorithm δ-LLL terminates within $\mathcal{P}oly(n)\cdot\widetilde{O}(\text{size}(C))$ bit operations. Let $(\boldsymbol{c}'_i)_{i\leq n}$ be the output basis. As size-reductions can be performed in time $\mathcal{P}oly(n)\cdot\widetilde{O}(\text{size}(C)))$, it suffices to show that the number of loop iterations of δ-LLL given $(\boldsymbol{c}_i)_{i\leq n}$ as input is $\mathcal{P}oly(n)$. To do this, it suffices to bound $\frac{\Pi((\boldsymbol{c}_i)_{i\leq n})}{\Pi((\boldsymbol{c}'_i)_{i\leq n})}$ by $2^{\mathcal{P}oly(n)}$.

First of all, we have $\lambda_1(L[(\boldsymbol{c}_j^{(i)})_{i\leq j\leq n}]) \leq \lambda_i(L)$, for all $i \leq n$. Indeed, let $\boldsymbol{v}_1,\ldots,\boldsymbol{v}_i \in L$ be linearly independent such that $\max_{j\leq i}\|\boldsymbol{v}_j\| \leq \lambda_i(L)$; at least one of them, say $\boldsymbol{v}_1$, remains non-zero when projected orthogonally to Span $(\boldsymbol{c}_j)_{j<i}$. We thus have $\lambda_1(L[(\boldsymbol{c}_j^{(i)})_{i\leq j\leq n}]) \leq \|\boldsymbol{v}_1\| \leq \lambda_i(L)$. Now, using (5), we obtain:

$$\Pi((\boldsymbol{c}_i)_{i\leq n}) = \prod_{i=1}^{n}\|\boldsymbol{c}_i^*\|^{2(n-i+1)} \leq 2^{O(n^3)}\prod_{i=1}^{n}\lambda_i(L)^{2(n-i+1)}.$$

On the other hand, we have (see [16, (1.7)]) $\lambda_i(L) \leq \max_{j\leq i}\|\boldsymbol{c}'_j\| \leq (\frac{1}{\sqrt{\delta-1/4}})^{i-1}\|\boldsymbol{c}'^*_i\|$, for all $i \leq n$. As a consequence, we have $\Pi((\boldsymbol{c}'_i)_{i\leq n}) \geq 2^{-O(n^3)}\cdot\prod_{i=1}^{n}\lambda_i(L)^{2(n-i+1)}$. This completes the proof. □

Acknowledgments. We thank N. Gama and P. Q. Nguyen for explaining to us their bound on the number of tours of the original BKZ algorithm. We also thank C.-P. Schnorr for helpful discussions. The authors were partly supported by the LaRedA ANR grant and an ARC Discovery Grant DP110100628.

References

1. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Proc. of STOC, pp. 99–108. ACM, New York (1996)
2. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. In: Proc. of STOC, pp. 601–610. ACM, New York (2001)
3. Akhavi, A.: Worst-case complexity of the optimal LLL algorithm. In: Gonnet, G.H., Viola, A. (eds.) LATIN 2000. LNCS, vol. 1776, pp. 355–366. Springer, Heidelberg (2000)
4. Cadé, D., Pujol, X., Stehlé, D.: fplll-3.1, a floating-point LLL implementation, `http://perso.ens-lyon.fr/damien.stehle`

5. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology 10(4), 233–260 (1997)
6. Gama, N., Howgrave-Graham, N., Koy, H., Nguyên, P.Q.: Rankin's constant and blockwise lattice reduction. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 112–130. Springer, Heidelberg (2006)
7. Gama, N., Nguyen, P.Q.: Finding short lattice vectors within Mordell's inequality. In: Proc. of STOC, pp. 207–216. ACM, New York (2008)
8. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008)
9. Goldreich, O., Goldwasser, S., Halevi, S.: Collision-free hashing from lattice problems. TR96-056 (1996), http://www.eccc.uni-trier.de/
10. Hanrot, G., Pujol, X., Stehlé, D.: Terminating BKZ. Cryptology ePrint Archive (2011), http://eprint.iacr.org/2011/198
11. Hanrot, G., Stehlé, D.: Improved analysis of Kannan's shortest lattice vector algorithm (extended abstract). In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 170–186. Springer, Heidelberg (2007)
12. Haviv, I., Regev, O.: Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In: Proc. of STOC, pp. 469–477. ACM, New York (2007)
13. Hirschhorn, P.S., Hoffstein, J., Howgrave-Graham, N., Whyte, W.: Choosing NTRUEncrypt parameters in light of combined lattice reduction and MITM approaches. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 437–455. Springer, Heidelberg (2009)
14. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998)
15. Kannan, R.: Improved algorithms for integer programming and related lattice problems. In: Proc. of STOC, pp. 99–108. ACM, New York (1983)
16. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Math. Ann. 261, 515–534 (1982)
17. Lenstra Jr., H.W.: Flags and lattice basis reduction. In: Proceedings of the Third European Congress of Mathematics, vol. 1. Birkhäuser, Basel (2001) **1**
18. Lovász, L.: An Algorithmic Theory of Numbers, Graphs and Convexity. CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, Philadelphia (1986)
19. Madritsch, M. G., Vallée, B.: Modelling the LLL algorithm by sandpiles. In: López-Ortiz, A. (ed.) LATIN 2010. LNCS, vol. 6034, pp. 267–281. Springer, Heidelberg (2010)
20. Martinet, J.: Perfect Lattices in Euclidean Spaces. Springer, Heidelberg (2002)
21. Micciancio, D., Regev, O.: Lattice-based cryptography. In: Bernstein, D.J., Buchmann, J., Dahmen, E. (eds.) Post-Quantum Cryptography, pp. 147–191. Springer, Heidelberg (2009)
22. Micciancio, D., Voulgaris, P.: A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. In: Proc. of STOC, pp. 351–358. ACM, New York (2010)
23. Micciancio, D., Voulgaris, P.: Faster exponential time algorithms for the shortest vector problem. In: Proc. of SODA. ACM, New York (2010)
24. Nguyên, P.Q.: Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto'97. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 288–304. Springer, Heidelberg (1999)

25. Nguyên, P.Q., Stehlé, D.: LLL on the average. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 238–256. Springer, Heidelberg (2006)
26. Nguyen, P.Q., Stehlé, D.: An LLL algorithm with quadratic complexity. SIAM J. Comput. 39(3), 874–903 (2009)
27. Nguyên, P.Q., Stern, J.: The two faces of lattices in cryptology. In: Silverman, J.H. (ed.) CaLC 2001. LNCS, vol. 2146, pp. 146–180. Springer, Heidelberg (2001)
28. Nguyen, P.Q., Vallée, B. (eds.): The LLL Algorithm: Survey and Applications. Information Security and Cryptography. Springer, Heidelberg (2009)
29. Novocin, A., Stehlé, D., Villard, G.: An LLL-reduction algorithm with quasi-linear time complexity. To Appear in the Proceedings of STOC (2011), http://prunel.ccsd.cnrs.fr/ensl-00534899/en
30. Pujol, X., Stehlé, D.: Rigorous and efficient short lattice vectors enumeration. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 390–405. Springer, Heidelberg (2008)
31. Regev, O.: The learning with errors problem. In: Invited Survey in CCC 2010 (2010), http://www.cs.tau.ac.il/~odedr/
32. Schnorr, C.P.: Progress on LLL and lattice reduction. In: [28]
33. Schnorr, C.P.: A hierarchy of polynomial lattice basis reduction algorithms. Theor. Comput. Science 53, 201–224 (1987)
34. Schnorr, C.P.: Block reduced lattice bases and successive minima. Combinatorics, Probability and Computing 3, 507–533 (1994)
35. Schnorr, C.P.: Accelerated slide- and LLL-reduction. Electronic Colloquium on Computational Complexity (ECCC) 11(50) (2011)
36. Schnorr, C.P., Euchner, M.: Lattice basis reduction: Improved practical algorithms and solving subset sum problems. In: Budach, L. (ed.) FCT 1991. LNCS, vol. 529, pp. 68–85. Springer, Heidelberg (1991)
37. Schnorr, C.P., Euchner, M.: Lattice basis reduction: improved practical algorithms and solving subset sum problems. Mathematics of Programming 66, 181–199 (1994)
38. Schnorr, C.P., Hörner, H.H.: Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 1–12. Springer, Heidelberg (1995)
39. Schönhage, A.: Fast reduction and composition of binary quadratic forms. In: Proceedings of the 1991 International Symposium on Symbolic and Algebraic Computation (ISSAC 1991), pp. 128–133. ACM, New York (1991)
40. Shoup, V.: NTL, Number Theory C++ Library, http://www.shoup.net/ntl/
41. Yap, C.K.: Fast unimodular reduction: planar integer lattices. In: Proceedings of the 1992 Symposium on the Foundations of Computer Science (FOCS 1992), pp. 437–446. IEEE Computer Society Press, Los Alamitos (1992)