Lattice Point Enumeration on Block Reduced Bases[*]

Michael Walter
UCSD
`miwalter@eng.ucsd.edu`

Abstract

When analyzing lattice-based cryptosystems, we often need to solve the Shortest Vector Problem (SVP) in some lattice associated to the system under scrutiny. The go-to algorithms in practice to solve SVP are enumeration algorithms, which usually consist of a preprocessing step, followed by an exhaustive search. Obviously, the two steps offer a trade-off and should be balanced in their running time in order to minimize the overall complexity. In practice, the most common approach to control this trade-off is to use block reduction algorithms during the preprocessing. Despite the popularity of this approach, it lacks any well founded analysis and all practical approaches seem to use ad hoc parameters. This weakens our confidence in the cryptanalysis of the systems. In this work, we aim to shed light on at least one side of this trade-off and analyze the effect of block reduction on the exhaustive search. For this, we give asymptotic worst case bounds and present results from both experiments and simulation that show its average case behavior in practice.

1 Introduction

Lattice-based cryptography is a very active research area having attracted a lot of attention in recent years. There are several reasons for this. On the one hand, lattice problems seem very useful in the construction of new cryptographic primitives. For example, they have been used to construct candidates for fully homomorphic encryption [13] and multi-linear maps [12], making significant progress on long standing open problems in cryptography. Furthermore, since Ajtai showed in his breakthrough work [2] a worst-case to average-case reduction for lattice problems, many primitives have emerged that enjoy a strong security reduction, see for example [23, 3, 28, 21, 22, 14]. Finally, lattices have been studied in mathematics and computer science for a long time and the hardness of many associated problems is well understood by now. One of the most classical and prominent problems of this kind is the Shortest Vector Problem (SVP). The SVP has been proved to be NP-hard under randomized reductions [19, 24], and thus all known algorithms to solve it have a (super) exponential complexity. Unfortunately, these algorithms are not very well understood. This is demonstrated by the recurring phenomenon that, as already pointed out in [10], asymptotically fast algorithms are routinely outperformed in practice by algorithms having inferior theoretical bounds. One such example are block reduction algorithms (which actually approximate SVP, rather than solve it). In theory, slide reduction [10] seems to offer the best trade-off between

[*]Research supported in part by the DARPA PROCEED program and NSF grant CNS-1117936. Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA or NSF.

running time and output quality, but is outperformed in practice by BKZ [29], as demonstrated in [11]. Such gaps between theory and practice weaken our confidence in the cryptanalysis of lattice based cryptosystems and thus hinder a wide spread adoption in practice.

Another example for such a gap are exact SVP algorithms. Sieving [4, 6, 26] and Voronoi cell based algorithms [25] are known to have a single exponential complexity, but they are hardly ever used to solve SVP in practice, since they are outperformed by enumeration for all practically feasible instances (see e.g. [27]), which is only known to have super exponential complexity in the worst case. Even inside the class of enumeration algorithms, such gaps exist. For example, for a long time, we only knew two kinds of enumeration algorithms that solve SVP: FinckePohst [9] (including a wide range of heuristic variants) having a worst case complexity of $2^{O(n^2)}$, and Kannan's algorithm [18] with a complexity of $n^{O(n)}$. But again, the latter is outperformed by the former in practice.

All known enumeration algorithms consist of two phases: a preprocessing that prepares the input, followed by an exhaustive search for the shortest vector. Roughly speaking, the exhaustive search is the source of the inefficiency for these algorithms and the asymptotic superiority of Kannan's algorithm stems from a very heavy preprocessing, significantly reducing the search space. However, it is also exactly this preprocessing that slows it down in practice, even though asymptotically it is dominated by the search step. Only very recently it was shown how to reduce this preprocessing step while keeping the search efficient [27]. On the other hand, it was already pointed out in [17], that spending more time on the preprocessing than the FinckePohst algorithm also makes sense in practice. To demonstrate this, a block reduction algorithm was used as the preprocessing to speed up the exhaustive search. The parameter of the reduction algorithm that controls the quality of the output can be used to trade off preprocessing time for a faster exhaustive search. A similar approach is also used in [7] and represents the state of the art. Despite the popularity of this method, to the best of our knowledge there is no well founded analysis of the trade-off that can be achieved, and all practical approaches seem to use ad hoc parameters.

Contribution In this work, we aim to shed light on at least one side of the trade-off between preprocessing and exhaustive search in enumeration algorithms by obtaining explicit asymptotic bounds for the exhaustive search depending on the kind and parameter of the approximation algorithm used as preprocessing. We first show what a straight forward analysis might look like. While this already shows that the enumeration complexity drops from $2^{O(n^2)}$ to $2^{O((n^2 \log^2 \beta)/\beta)}$ when preprocessing the basis with block size β, we will argue that this bound is too rough by drawing a connection between enumeration on block reduced bases with large block size and Kannan's algorithm. We then build on techniques from [27] to show that the enumeration complexity in fact drops to $\beta^{O(n^2/\beta)}$. We also hint at implications these bounds have for the complexity of the entire algorithm, but leave an in depth analysis for future work. Instead, we move on to the practical side and show through a number of experiments and simulations that in practice the exhaustive search has a complexity qualitatively similar to the asymptotic bounds. We hope that this work helps to further our understanding of lattice algorithms and to ultimately lead to a wider adoption of lattice-based cryptography in practice.

2 Preliminaries

Notation Numbers and reals are denoted by lower case letters and sets by upper case letters. For $n \in \mathbb{Z}_+$ we denote the set $\{1, \ldots, n\}$ by $[n]$. For vectors we use bold lower case letters and the i-th entry of a vector $\mathbf{v}$ is denoted by v_i. Let $\langle \mathbf{v}, \mathbf{w} \rangle = \sum_i v_i \cdot w_i$ be the scalar product of two vectors, and $\|\mathbf{v}\| = \sqrt{\langle \mathbf{v}, \mathbf{v} \rangle}$ the standard Euclidean norm. We define the projection of a vector $\mathbf{b}$ orthogonally to a vector $\mathbf{v}$ as $\pi_{\mathbf{v}}(\mathbf{b}) = \mathbf{b} - \frac{\langle \mathbf{b}, \mathbf{v} \rangle}{\|\mathbf{v}\|^2}\mathbf{v}$. Matrices are denoted by bold upper case letters. The i-th column of a matrix $\mathbf{B}$ is denoted by $\mathbf{b}_i$. Furthermore, we denote the submatrix comprising the columns from the i-th to the j-th column (inclusive) as $\mathbf{B}_{[i,j]}$. We extend the projection operator to matrices, where $\pi_{\mathbf{V}}(\mathbf{B})$ is the matrix obtained by applying $\pi_{\mathbf{V}}$ to every column $\mathbf{b}_i$ of $\mathbf{B}$ and $\pi_{\mathbf{V}}(\mathbf{b}_i) = \pi_{\mathbf{v}_k}(\cdots(\pi_{\mathbf{v}_1}(\mathbf{b}_i))\cdots)$.

Lattices A *lattice* Λ is a discrete subgroup of $\mathbb{R}^m$ and is generated by a matrix $\mathbf{B} \in \mathbb{R}^{m \times n}$, i.e. $\Lambda = \mathcal{L}(\mathbf{B}) = \{\mathbf{B}\mathbf{x} \colon \mathbf{x} \in \mathbb{Z}^n\}$. If $\mathbf{B}$ has full column rank, it is called a *basis* of Λ and $\dim(\Lambda) = n$ is the dimension (or rank) of Λ. A lattice has infinitely many bases, which are related to each other by right-multiplication with unimodular matrices. With each matrix $\mathbf{B}$ we associate its *Gram-Schmidt-Orthogonalization* (GSO) $\mathbf{B}^*$, where the i-th column $\mathbf{b}_i^*$ of $\mathbf{B}^*$ is defined as $\mathbf{b}_i^* = \pi_{\mathbf{B}_{[1,i-1]}}(\mathbf{b}_i) = \pi_{\mathbf{B}^*_{[1,i-1]}}(\mathbf{b}_i)$ (and $\mathbf{b}_1^* = \mathbf{b}_1$). For a fixed matrix $\mathbf{B}$ we extend the projection operation to indices: $\pi_i(\cdot) = \pi_{\mathbf{B}^*_{[1,i]}}(\cdot)$. Whenever we refer to the *shape* of a basis $\mathbf{B}$, we mean the vector $(\|\mathbf{b}_i^*\|)_{i \in [n]}$.

For every lattice Λ there are a few invariants associated to it. One of them is its determinant $\det(\mathcal{L}(\mathbf{B})) = \prod_i \|\mathbf{b}_i^*\|$ for any basis $\mathbf{B}$. Even though the basis of a lattice is not uniquely defined, the determinant is and it is efficiently computable given a basis. Furthermore, for every lattice Λ we denote the length of its shortest non-zero vector (also known as the *first minimum*) by $\lambda_1(\Lambda)$, which is always well defined. We use the short-hand notations $\det(\mathbf{B}) = \det(\mathcal{L}(\mathbf{B}))$ and $\lambda_1(\mathbf{B}) = \lambda_1(\mathcal{L}(\mathbf{B}))$. Minkowski's theorem is a classic result that relates the first minimum to the determinant of a lattice. It states that $\lambda_1(\Lambda) \leq \sqrt{\gamma_n} \det(\Lambda)^{1/n}$, for any Λ with $\dim(\Lambda) = n$, where $\Omega(n) \leq \gamma_n \leq n$ is Hermite's constant. Finding a (even approximate) shortest nonzero vector in a lattice, commonly known as the *Shortest Vector Problem* (SVP), is NP-hard under randomized reductions [19, 24].

Lattice Reduction Lattice reduction algorithms deal with the problem of obtaining a "good" basis from an arbitrary basis for some notion of a "good" basis. The *LLL* algorithm [20] is a polynomial time basis reduction algorithm that produces a basis $\mathbf{B} \in \mathbb{Z}^{m \times n}$ such that $\delta\|\mathbf{b}_i^*\| \leq \lambda_1(\pi_{i-1}(\mathbf{B}_{[i,i+1]}))$ for all $i \in [n-1]$ and some $\delta < 1$ usually chosen close to 1.

BKZ-β [29] is a generalization of LLL to larger block size, i.e. it guarantees that the output basis satisfies $\delta\|\mathbf{b}_i^*\| \leq \lambda_1(\pi_{i-1}(\mathbf{B}_{[i,\min(i+\beta,n)]}))$ for all $i \in [n-1]$ by utilizing a SVP oracle in dimension β. When $\beta = n$, this is usually referred to as *HKZ* reduction and is essentially equivalent to solving SVP. Using Minkowski's theorem, one can prove the following bounds for $\mathbf{b}_1$ of a BKZ-β reduced basis [16]:

$$\|\mathbf{b}_1\| \leq \beta^{\frac{n-1}{\beta-1}} \lambda_1(\mathbf{B}) \tag{1}$$

$$\|\mathbf{b}_1\| \leq \beta^{\frac{n-1}{2(\beta-1)}+\frac{3}{2}} \det(\mathbf{B})^{1/n} \tag{2}$$

Note that any prefix $\mathbf{B}_{[1,i]}$ and any projection $\pi_i(\mathbf{B})$ of a BKZ-β reduced basis is also BKZ-β reduced. Unfortunately, there is no polynomial bound on the number of calls BKZ makes to the SVP oracle, but it has been repeatedly reported to behave very well in practice (see e.g. [11, 7]). Furthermore, Hanrot, Pujol, and Stehlé showed in [16] that one can terminate BKZ after a polynomial number of calls to the SVP oracle and provably achieve bounds only slightly worse than (1). For these reasons, BKZ is very popular in practice and implementations are readily available in different libraries, e.g. in `NTL`[31] or `fpLLL`[5]. As the dimension and block size of BKZ grows, running it becomes more and more impractical. But since BKZ has also proved to be a very useful tool in the cryptanalysis of lattice-based cryptosystems, one would like to predict its behavior for very large instances to estimate the security of such systems. To this end, Chen and Nguyen introduced a BKZ simulator [7] that, given as input the shape of a basis and an integer closely related to the number of SVP calls, predicts the shape of the output of BKZ after the given number of calls to the oracle without the need to run it, based on heuristic assumptions. It is straightforward to modify the simulator to predict the output of BKZ by calling it repeatedly until no more change to the shape of the basis is observed.

In [10], Gama and Nguyen introduced a different block reduction algorithm, namely *slide reduction*. Similar to BKZ, it is parameterized by a block size β and uses a SVP oracle in dimension β to produce a basis with the following properties:

$$\|\mathbf{b}_1\| \leq \beta^{\frac{n-\beta}{\beta-1}} \lambda_1(\mathbf{B}) \tag{3}$$

$$\|\mathbf{b}_1\| \leq \beta^{\frac{n-1}{2(\beta-1)}} \det(\mathbf{B})^{1/n} \tag{4}$$

Moreover, every prefix $\mathbf{B}_{[1,i\beta]}$ and every projection $\pi_{i\beta}(\mathbf{B})$ is slide reduced and every projected block $\pi_{i\beta}(\mathbf{B}_{[i\beta+1,(i+1)\beta]})$ is HKZ reduced. Slide reduction has the desirable property of only making a polynomial number of calls to the SVP oracle. Unfortunately, as reported in [10] and [11], it seems to be outperformed by BKZ, despite providing better guarantees on output quality and runtime. Not surprisingly, it is rarely used in practice and we are not aware of any publicly available implementation.

Enumeration Algorithms The standard enumeration procedure, usually attributed to Fincke, Pohst [9], and Kannan [18] can be described as a recursive algorithm: given as input a basis $\mathbf{B} \in \mathbb{Z}^{m\times n}$ and a radius r, it first recursively finds all vectors $\mathbf{v}' \in \Lambda(\pi_1(\mathbf{B}))$ with $\|\mathbf{v}'\| \leq r$, and then for each of them finds all $\mathbf{v} \in \Lambda(\mathbf{B})$, s.t. $\pi_1(\mathbf{v}) = \mathbf{v}'$ and $\|\mathbf{v}\| \leq r$, using $\mathbf{b}_1$. For each $i \in [n]$, the procedure introduces a multiplicative factor proportional to $r/\|\mathbf{b}_i^*\|$ to its complexity. So the complexity of the enumeration procedure depends on a) the upper bound r for the shortest vector and b) the shape of the basis. In fact, Hanrot and Stehlé noticed in [17] that one can estimate the complexity of enumeration based on the Gaussian heuristic by the quantity

$$E(r,\mathbf{B}) = \max_{i\in[n]} \frac{\pi^{i/2} r^i}{\Gamma(i/2+1)\prod_{j\geq n-i+1}\|\mathbf{b}_i^*\|} \tag{5}$$

We remark that, as already pointed out in [7], this estimate is likely to overestimate the complexity, since it does note take heuristics like dynamic radius updates etc. into account. So Equation (5) should not be used as a precise prediction, but it is very useful to compare the expected complexity of enumeration for different inputs.

The bound r is usually chosen to be either the length of first vector $\|\mathbf{b}_1\|$ of the basis or Minkowski's bound. The shape of the basis is determined by the preprocessing strategy and there is a trade-off between preprocessing the basis and enumeration. The simplest approach, first proposed by Fincke and Pohst [9], is to apply LLL to the input and then call the enumeration procedure. This algorithm has a worst case complexity of $2^{O(n^2)}$. On the other hand, Kannan [18] proposed to use a much heavier, recursive preprocessing: alternate LLL reduction on $\mathbf{B}_{[1,2]}$ and recurse on $\pi_1(\mathbf{B})$ until no more change is observed. At this point the basis is often called *quasi-HKZ reduced.* Only then call the enumeration procedure to find the shortest vector $\mathbf{v}$ and finally recurse on $\pi_{\mathbf{v}}(\mathbf{B})$ to fully HKZ reduce the basis (which is necessary for the recursive calls to make sense). It can be shown that this algorithm runs in at most $O(n^{n/2e+o(n)})$ steps [17]. In theory, this is much better than the bound obtained for FinckePohst, but the heavy preprocessing seems to kill the performance in practice, so it is never used. It was only very recently, that a technique for interpolating both algorithms was introduced [27], providing an easy method to trade off preprocessing time and enumeration complexity. In practice, the most common approach at this point is to use block reduction algorithms to preprocess the basis before enumeration (see e.g. [17, 7]), but to the best of our knowledge there is no analysis of this approach and the parameter choice is usually ad hoc.

3 Worst-Case Analysis of Enumeration after Block Reduction

Consider an algorithm that reduces an input basis $\mathbf{B}$ using a block reduction algorithm with parameter β and then runs enumeration to find the shortest vector. We are interested in the complexity of the enumeration step depending on the parameter β.

3.1 A Naive Attempt

As a warm up we will show how a simple analysis could go and argue why this result is not satisfactory. Assume for now that the enumeration bound is chosen to be $r = \|\mathbf{b}_1\|$ and that $\|\mathbf{b}_1\| \gtrsim \|\mathbf{b}_i^*\|$ for all $i \in [n]$ (which is true under a common heuristic assumption, namely the Geometric Series Assumption [30]). In this case, enumeration can be bounded by the quantity

$$\prod_i \frac{\|\mathbf{b}_1\|}{\|\mathbf{b}_i^*\|}$$

One can easily show (see e.g. [8], Theorem 1) for BKZ-β reduced bases that

$$\|\mathbf{b}_1\| \leq \beta^{\frac{\log\beta+1}{2}(1+\frac{i-1}{\beta-1})}\|\mathbf{b}_i^*\| \approx 2^{(i\log^2\beta)/2\beta}\|\mathbf{b}_i^*\|$$

which results in a bound for enumeration of $2^{(n^2/4\beta)\log^2\beta}$. While already showing that there is at least an improvement of $(\log^2\beta)/\beta$ as compared to FinckePohst (which corresponds to the case of $\beta = 2$), this bound seems to be too rough. Consider the algorithm with parameter $\beta = n-1$. In this case, it has a striking similarity to Kannan's algorithm: Recall that Kannan's algorithm can be viewed as alternately calling a SVP oracle on $\mathbf{B}_{[1,2]}$ and a HKZ oracle (instantiated with a recursive call) on $\pi_1(\mathbf{B})$. The only difference between this kind of preprocessing and BKZ-$(n-1)$ is that the latter alternates between calling an SVP oracle on $\mathbf{B}_{[1,n-1]}$ and a HKZ oracle on $\pi_1(\mathbf{B})$. So a BKZ-$(n-1)$ reduced basis is also quasi-HKZ reduced and thus strictly stronger reduced than after Kannan's preprocessing. In particular, the enumeration complexity should not be larger than for Kannan's algorithm, which we know to be $O(n^{n/2e+o(n)})$. However, plugging $\beta = n-1$ into the bound we obtained above, we see that the complexity is bounded by $n^{O(n\log n)}$, which is off by a factor $\log n$ in the exponent. It follows that the analysis could be improved by at least that factor.

We remark that we do not believe that BKZ-$(n-1)$ is a suitable preprocessing for enumeration since it is even more expensive than quasi-HKZ reduction and can be expected to be at least as impractical.

3.2 ζ-Reducedness of Block Reduced Bases

Our analysis of the enumeration builds on the framework recently introduced in [27], namely ζ-reduction. We recall the corresponding definition.

Definition 1 ([27]) *Let $\mathbf{B} \in \mathbb{Z}^{m \times n}$ be a lattice basis*[1] *and $\zeta : [n] \to \mathbb{R}_+$. We call $\mathbf{B}$* ζ-reduced*, if for all $i \in [n]$*

$$\|\mathbf{b}_i^*\| > \zeta(i) \det(\mathbf{B})^{1/n} \quad \Rightarrow \quad \lambda_1(\pi_{i-1}(\mathbf{B})) > \lambda_1(\mathbf{B})$$

and $\mathbf{B}_{[1,k]}$ is ζ-reduced for all $k \in [n-1]$.

In [27] it was proved that using the ζ-reducedness of a basis we can bound the enumeration step:

Theorem 1 ([27]) *Let $\mathbf{B} \in \mathbb{Z}^{m \times n}$ be a ζ-reduced basis with an efficiently computable $\zeta(i) \geq \sqrt{n}$ for all $i \in [n]$. Then there is an efficiently computable set $M \subset \mathbb{Z}^n$ with $|M| \leq 3^n \prod_{i=1}^n \zeta(i)$ such that there is a vector $\mathbf{x} \in M$ with $\|\mathbf{B}\mathbf{x}\| = \lambda_1(\mathbf{B})$.*

It follows that in order to bound the enumeration on BKZ reduced bases, it suffices to analyze the ζ bounds that BKZ achieves. This is exactly what the following lemma does.

Lemma 1 *If $\mathbf{B} \in \mathbb{Z}^{m \times n}$ is BKZ-β reduced then it is ζ-reduced with $\zeta(i) = \beta^{\frac{n-1}{2(\beta-1)} + \frac{3}{2}}$.*

Proof We prove the contrapositive and assume $\lambda_1(\pi_{i-1}(\mathbf{B})) \leq \lambda_1(\mathbf{B})$. Since $\pi_{i-1}(\mathbf{B})$ and $\mathbf{B}_{[1,i-1]}$ are BKZ-β reduced, we have

$$\begin{aligned}
\|\mathbf{b}_i^*\| \leq& \beta^{\frac{n-i}{(\beta-1)}} \lambda_1(\pi_{i-1}(\mathbf{B})) \\
\leq& \beta^{\frac{n-i}{(\beta-1)}} \lambda_1(\mathbf{B}) \\
\leq& \beta^{\frac{n-i}{(\beta-1)}} \|\mathbf{b}_1\| \\
\leq& \beta^{\frac{n-i}{(\beta-1)}} \beta^{\frac{i-2}{2(\beta-1)} + \frac{3}{2}} \det(\mathbf{B}_{[1,i-1]})^{1/(i-1)}
\end{aligned}$$

and so

$$\|\mathbf{b}_i^*\|^{i-1} \leq \beta^{\frac{(i-1)(n-i)}{(\beta-1)} + \frac{(i-1)(i-2)}{2(\beta-1)} + \frac{3}{2}(i-1)} \det(\mathbf{B}_{[1,i-1]})$$

By (2) we also have $\|\mathbf{b}_i^*\|^{n-i+1} \leq \beta^{\frac{(n-i)(n-i+1)}{2(\beta-1)} + \frac{3}{2}(n-i+1)} \det(\pi_{i-1}(\mathbf{B}))$. Multiplying those two bounds and doing some arithmetic gives

$$\|\mathbf{b}_i^*\|^n \leq \beta^{\frac{(i-1)(n-i)}{(\beta-1)} + \frac{(i-1)(i-2)}{2(\beta-1)} + \frac{(n-i)(n-i+1)}{2(\beta-1)} + \frac{3}{2}n} \det(\mathbf{B}) \leq \beta^{\frac{n(n-1)}{2(\beta-1)} + \frac{3}{2}n} \det(\mathbf{B})$$

□

Using Theorem 1 we can easily deduce a runtime bound for the enumeration step.

[1] In [27] the definition covers arbitrary generating systems, not just bases. In this work however, we only consider bases, so we slightly simplified the definition accordingly.

Corollary 1 *Given a BKZ-β reduced basis $\mathbf{B} \in \mathbb{Z}^{m \times n}$, enumeration can solve SVP in $\Lambda(\mathbf{B})$ in $\beta^{\frac{n(n-1)}{2(\beta-1)}+\frac{3}{2}n}2^{O(n)}$.*

For $\beta = 2$ (FinckePohst) and $\beta = n-1$ ($\approx$ Kannan) we get the expected bounds up to constants in the exponent. Other values for β interpolate the two algorithms offering an improvement in the exponent of the dominating factor of FinckePohst of about $\log(\beta)/\beta$. Up to constants in the exponent, this proves that the enumeration step after BKZ is as efficient as after Kannan's preprocessing as long as $\beta = O(n)$.

Although rarely used in practice, we now also show how to apply ζ-reduction to slide reduced bases, which leads to slightly improved results.

Lemma 2 *If $\mathbf{B} \in \mathbb{Z}^{m \times n}$ is slide reduced with parameter β the k-th projected block $\pi_{k\beta}(\mathbf{B}_{[k\beta+1,(k+1)\beta]})$ is ζ-reduced for*

$$\zeta(k\beta < i \leq (k+1)\beta) = \beta^{\frac{n-1}{2(\beta-1)}+\frac{\beta}{\beta-1}-\frac{k\beta^2}{n(\beta-1)}}$$

Proof As before, we assume $\lambda_1(\pi_{i-1}(\mathbf{B})) \leq \lambda_1(\mathbf{B})$. We start by showing the lemma for the first vector $\|\mathbf{b}_i^*\|$ of the block. In this case $\mathbf{B}_{[1,i-1]}$ and $\pi_{i-1}(\mathbf{B})$ are also slide reduced and we can apply the same approach as for BKZ:

$$\begin{aligned}\|\mathbf{b}_i^*\| &\leq \beta^{\frac{n-i-\beta}{(\beta-1)}} \lambda_1(\pi_{i-1}(\mathbf{B})) \\ &\leq \beta^{\frac{n-i-\beta}{(\beta-1)}} \lambda_1(\mathbf{B}) \\ &\leq \beta^{\frac{n-i-\beta}{(\beta-1)}} \|\mathbf{b}_1\| \\ &\leq \beta^{\frac{n-i-\beta}{(\beta-1)}} \beta^{\frac{i-2}{2(\beta-1)}} \det(\mathbf{B}_{[1,i-1]})^{1/(i-1)}\end{aligned}$$

and so

$$\|\mathbf{b}_i^*\|^{i-1} \leq \beta^{\frac{(i-1)(n-i-\beta)}{(\beta-1)}+\frac{(i-1)(i-2)}{2(\beta-1)}} \det(\mathbf{B}_{[1,i-1]})$$

By (4) we also have $\|\mathbf{b}_i^*\|^{n-i+1} \leq \beta^{\frac{(n-i)(n-i+1)}{2(\beta-1)}} \det(\pi_{i-1}(\mathbf{B}))$. Again, multiplying those two bounds gives

$$\begin{aligned}\|\mathbf{b}_i^*\|^n &\leq \beta^{\frac{(i-1)(n-i-\beta)}{(\beta-1)}+\frac{(i-1)(i-2)}{2(\beta-1)}+\frac{(n-i)(n-i+1)}{2(\beta-1)}} \det(\mathbf{B}) \\ &\leq \beta^{\frac{n(n-1)-2\beta(i-1)}{2(\beta-1)}} \det(\mathbf{B})\end{aligned} \tag{6}$$

which implies $\|\mathbf{b}_i^*\| \leq \beta^{\frac{n-1}{2(\beta-1)}} \det(\mathbf{B})^{1/n}$ and shows the result for the first vector of each block, because $\frac{\beta}{\beta-1} \geq \frac{k\beta^2}{n(\beta-1)}$ and so $\zeta(i) \geq \beta^{\frac{n-1}{2(\beta-1)}}$.

We now generalize to arbitrary i. Let $j = (k+1)\beta$, i.e. the end of the block. If $\lambda_1(\pi_j(\mathbf{B})) > \lambda_1(\pi_{i-1}(\mathbf{B}))$ then the shortest vector in $\pi_{i-1}(\mathbf{B})$ is in $\pi_{i-1}(\mathbf{B}_{[i,j]})$ and $\|\mathbf{b}_i^*\| = \lambda_1(\pi_{i-1}(\mathbf{B}))$, because $\pi_{i-1}(\mathbf{B}_{[i,j]})$ is HKZ reduced. It follows that $\|\mathbf{b}_i^*\|$ is ζ-reduced for all $\zeta(i) \geq \sqrt{n} \leq \beta^{\frac{n-1}{2(\beta-1)}}$. Now let $\lambda_1(\pi_j(\mathbf{B})) \leq \lambda_1(\pi_{i-1}(\mathbf{B}))$. Then by assumption $\lambda_1(\pi_j(\mathbf{B})) \leq \lambda_1(\pi_{i-1}(\mathbf{B})) \leq \lambda_1(\mathbf{B})$, so (6) holds for $\mathbf{b}_{j+1}^*$. Utilizing the fact that $\pi_{i-1}(\mathbf{B}_{[i,j]})$ is HKZ reduced and $\pi_i(\mathbf{B}_{[i+1,j+1]})$ is DSVP reduced, we easily deduce by Minkowski's theorem that $\|\mathbf{b}_i^*\| \leq \kappa^{\frac{\kappa}{\kappa-1}} \|\mathbf{b}_{j+1}^*\|$ where $\kappa = j - i + 1$. Putting this and (6) together, we get:

$$\begin{aligned}\|\mathbf{b}_i^*\| \leq& \kappa^{\frac{\kappa}{\kappa-1}} \|\mathbf{b}_{j+1}^*\| \\ \leq& \beta^{\frac{\beta}{\beta-1}} \beta^{\frac{n(n-1)-2\beta j}{2n(\beta-1)}} \det(\mathbf{B})^{1/n} \\ \leq& \beta^{\frac{n-1}{2(\beta-1)} + \frac{\beta}{\beta-1} - \frac{k\beta^2}{n(\beta-1)}} \det(\mathbf{B})^{1/n}\end{aligned}$$

□

Again, using Theorem 1, we obtain a bound on the runtime of enumeration on slide reduced bases.

Corollary 2 *Given a β-slide reduced basis $\mathbf{B} \in \mathbb{Z}^{m \times n}$, enumeration can solve the SVP in $\Lambda(\mathbf{B})$ in $\beta^{\frac{n(n-1)}{2(\beta-1)} + \frac{(n-\beta)}{2}} 2^{O(n)}$.*

Proof The corollary follows from a short sequence of equations:

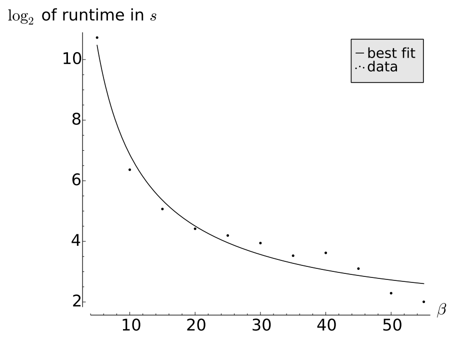
$$\begin{aligned}\prod_{k=1}^{n/\beta} \zeta((k-1)\beta+1)^\beta =& \beta^{\frac{n(n-1)}{2(\beta-1)} + \frac{n\beta}{\beta-1} - \frac{\beta^3}{n(\beta-1)} \sum_{k=1}^{n/\beta} k} \\ =& \beta^{\frac{n(n-1)}{2(\beta-1)} + \frac{n\beta}{\beta-1} - \frac{n\beta+\beta^2}{2(\beta-1)}} \\ =& \beta^{\frac{n(n-1)}{2(\beta-1)} + \frac{\beta(n-\beta)}{2(\beta-1)}} \approx \beta^{\frac{n(n-1)}{2(\beta-1)} + \frac{(n-\beta)}{2}}\end{aligned}$$

□

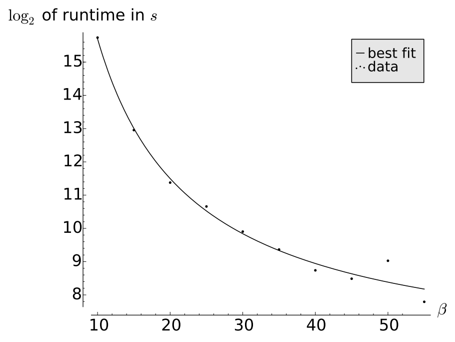
Not surprisingly, due to the better bounds achieved on $\|\mathbf{b}_1^*\|$, slide reduction yields a stronger ζ-reduction and thus improves the bound on the enumeration. However, plugging $\beta = n-1$ into the bound[2] shows that the bound is still worse than the one for Kannan, but only by a factor $1/e$. We leave it as an interesting open question if one can achieve such a bound for block reduced bases.

Remark Recall that block reduction algorithms use a SVP oracle in dimension β. Obviously, we can use recursive calls to our enumeration algorithm (including block reduction) to implement this oracle. In the case of BKZ we can use the slightly worse bound obtained in [16] instead of Equation (1). This will give us worse constants in the exponents, but has the advantage that the number of (top level) recursive calls during the preprocessing is polynomially bounded, which bounds the overall number of recursive calls by $n^{O(n)}$. This proves that using the algorithm proposed in [16] combined with ζ-reduction, SVP can be solved by block reduction and enumeration in $n^{O(n)}$ steps by setting $\beta = O(n)$. Alternatively, we can use slide reduction instead of BKZ to achieve a similar result. To the best of our knowledge, such a bound was only known for Kannan's algorithm and the recent variant in [27], up to this point.

[2]Technically, this choice of parameter is not possible for slide reduction as it requires $\beta | n$. But plugging in this value should give a good estimation of how tight the bound is by comparison with Kannan's algorithm.



(a) $\dim(\Lambda) = 75$



(b) $\dim(\Lambda) = 85$

Figure 1: Runtime of HKZ reduction after BKZ-β reduction

4 Performance in Practice

In this section we present experimental results that indicate that the bound obtained in Section 3 for BKZ is not only of theoretical nature, but that the average case hardness of enumeration on block reduced bases can be expected to follow similar bounds (with smaller constants) in practice.

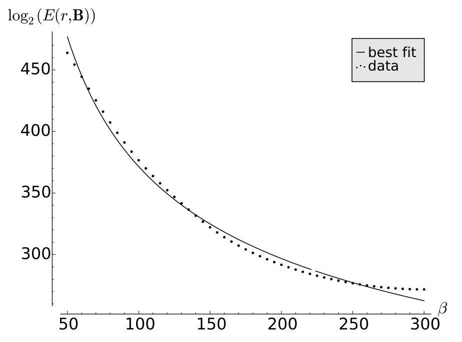
All experiments and simulations were performed on random lattices in the sense of Goldstein and Mayer [15] with numbers of bit length $10n$, where n is the lattice dimension. In order to demonstrate that the enumeration follows qualitatively similar bounds in practice, we use the model $\beta^{a\frac{n(n-1)}{(\beta-1)}+bn}2^{cn}$, where n is the lattice dimension, β the block size of BKZ, and a, b, and c are parameters. We fit the model to our data using standard statistical methods and show that it is indeed a good fit.

4.1 Experiments

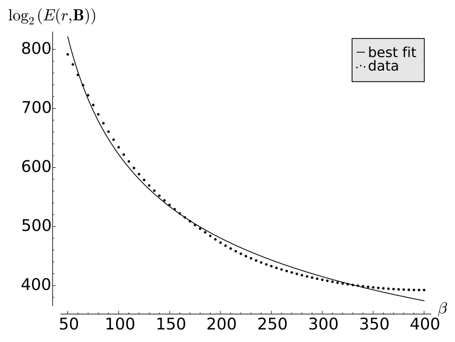
We used NTL's BKZ algorithm to reduce the input lattice of dimension n with varying parameter β, after which we called it again using $\beta = n$ to HKZ reduce the lattice (which is essentially equivalent to finding the shortest vector). We only measure the running time of the second call as the goal of this article is to explore the effect the reducedness has on the final enumeration. In order to obtain results in larger dimensions we set NTL's pruning parameter to 10. Still, the algorithm has its limits (both, the BKZ preprocessing and the final HKZ reduction) and we were only able to obtain meaningful results for $n \leq 85$ and $5 \leq \beta \leq 55$. We used the model for fixed n and fitted it to the data obtained by the experiments (where each data point is the average over 20 random lattices). Figure 1 shows exemplary results for $n = 75$ and $n = 85$, respectively. The results demonstrate that for fixed dimension n the running time of enumeration in practice closely follows the theoretical worst-case bound up to constants in the exponents. Curiously, the exponents for the closest curve vary for different dimensions - an issue, we will address later.

4.2 Simulation

As the experiments in the previous section are somewhat limited in scale due to resource constraints, we reverted to simulation to obtain data in larger dimensions. Specifically, we generated random lattices in dimension $n \in \{100, 110, \dots, 400\}$ and LLL reduced them using NTL. Then we used



(a) $\dim(\Lambda) = 300$



(b) $\dim(\Lambda) = 400$

Figure 2: Estimated runtime of enumeration (in nodes) after simulated BKZ-β reduction

the BKZ simulator[3] for each lattice to compute the expected shape of a BKZ-β reduced basis for $\beta \in \{50, 55, \ldots, n\}$. Finally, for each n we estimated the number of nodes that need to be enumerated using (5).

The result is shown in Figure 2. Again we see that the running time follows the theoretical bound and again we observed the phenomenon that the constants seem to depend on n. To explore this issue a little deeper, we fitted the model to the complete data set (i.e. now n and β are the variables). The result is shown in Figure 3 and indicates that for large dimensions the average case hardness can be expected to follow the model at least roughly. As we are mostly interested in the constant of the dominating term, we fixed b and c to the value obtained by this fitting and extracted the corresponding a parameter for each fixed n. Figure 4 plots the obtained a value depending on the dimension n. While the values do increase with the dimension, they do so less and less rapidly. From the theoretical analysis we know that they cannot increase indefinitely, so we conjecture that they converge at some point. We do not offer an explanation for this phenomenon and leave it for future work to explore this behavior in depth.

References

[1] ACM. *STOC 2008*, May 2008.

[2] M. Ajtai. Generating hard instances of lattice problems. *Complexity of Computations and Proofs, Quaderni di Matematica*, 13:1–32, 2004. Preliminary version in STOC 1996.

[3] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of STOC '97*, pages 284–293. ACM, May 1997.

[4] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings of STOC*, pages 266–275. ACM, July 2001.

[5] M. Albrecht, D. Cadé, X. Pujol, and D. Stehlé. fplll-4.0, a floating-point LLL implementation. Available at `http://perso.ens-lyon.fr/damien.stehle`.

[3]Our implementation of the simulator is available at `http://cseweb.ucsd.edu/~miwalter/src/sim_bkz.sage`

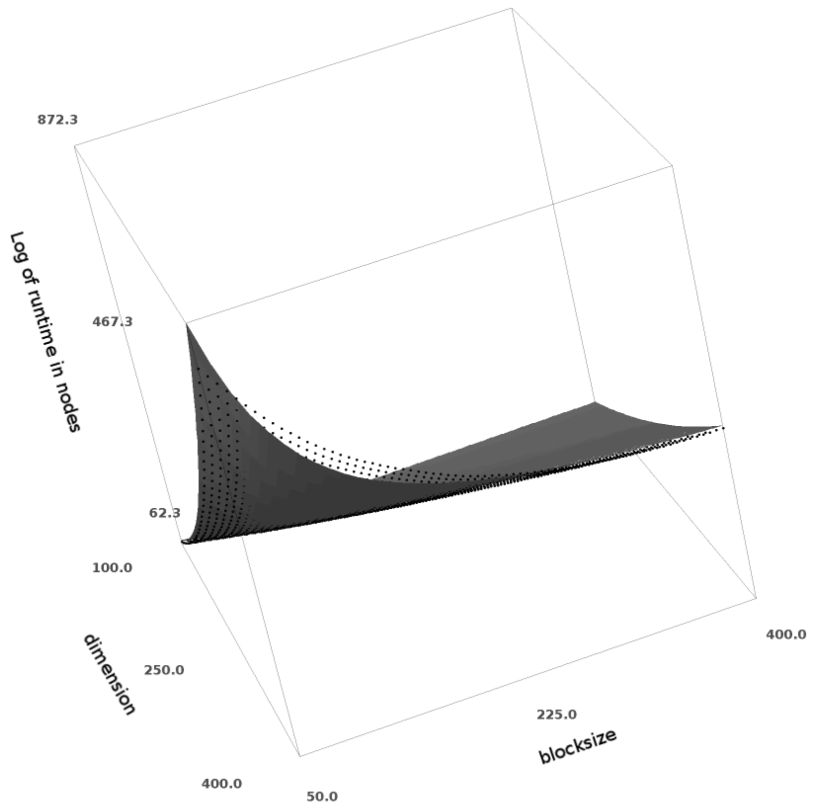


Figure 3: Log of runtime in nodes for full enumeration depending on dimension n and blocksize β

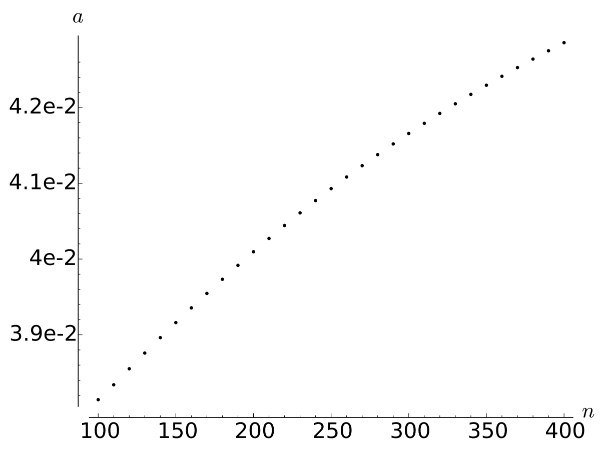


Figure 4: Fitted value for parameter a for different $n = \dim(\Lambda)$ and fixed parameters b and c

[6] J. Blömer and S. Naewe. Sampling methods for shortest vectors, closest vectors and successive minima. *Theoretical Computer Science*, 410(18):1648–1665, Apr. 2009. Preliminary version in ICALP 2007.

[7] Y. Chen and P. Q. Nguyen. Bkz 2.0: Better lattice security estimates. In *ASIACRYPT*, volume 7073 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2011.

[8] D. Ding, G. Zhu, and X. Wang. A genetic algorithm for searching shortest lattice vector of svp challenge. Cryptology ePrint Archive, Report 2014/489, 2014. `http://eprint.iacr.org/`.

[9] U. Fincke and M. Pohst. Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. *Mathematics of Computation*, 44:463–471, 1985.

[10] N. Gama and P. Q. Nguyen. Finding short lattice vectors within Mordell's inequality. In *Proceedings of STOC* [1], pages 207–216.

[11] N. Gama and P. Q. Nguyen. Predicting lattice reduction. In *Proceedings of Eurocrypt*, volume 4965 of *LNCS*, pages 31–51. Springer, 2008.

[12] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In T. Johansson and P. Nguyen, editors, *Advances in Cryptology EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17. Springer Berlin Heidelberg, 2013.

[13] C. Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of STOC*, pages 169–178. ACM, 2009.

[14] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of STOC* [1], pages 197–206.

[15] D. Goldstein and A. Mayer. On the equidistribution of Hecke points. *Forum Mathematicum*, 15(2):165 – 189, 2003.

[16] G. Hanrot, X. Pujol, and D. Stehlé. Terminating BKZ. Cryptology ePrint Archive, Report 2011/198, 2011. `http://eprint.iacr.org/`.

[17] G. Hanrot and D. Stehlé. Improved analysis of kannan's shortest lattice vector algorithm. In *Proceedings of Crypto*, volume 4622 of *LNCS*, pages 170–186. Springer, Aug. 2007.

[18] R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proceedings of the fifteenth annual ACM symposium on theory of computing - STOC '83*, pages 193–206. ACM, Apr. 1983. Journal version in Math. of Operation Research 12(3):415-440, (1987).

[19] S. Khot. Hardness of approximating the shortest vector problem in lattices. *Journal of the ACM*, 52(5):789–808, Sept. 2005. Preliminary version in FOCS 2004.

[20] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:513–534, 1982.

[21] R. Lindner and C. Peikert. Better key sizes (and attacks) for lwe-based encryption. In A. Kiayias, editor, *Topics in Cryptology CT-RSA 2011*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer, 2011.

[22] V. Lyubashevsky and D. Micciancio. Asymptotically efficient lattice-based digital signatures. In *Proceedings of TCC*, volume 4948 of *LNCS*, pages 37–54. Springer, Mar. 2008.

[23] V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. SWIFFT: a modest proposal for FFT hashing. In *Fast Software Encryption – Proceedings*, volume 5086 of *LNCS*, pages 54–72. Springer, Feb. 2008.

[24] D. Micciancio. Inapproximability of the shortest vector problem: Toward a deterministic reduction. *Theory of Computing*, 8(1):487–512, 2012.

[25] D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. In *Proceedings of STOC*, pages 351–358, 2010.

[26] D. Micciancio and P. Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proceedings of SODA*, pages 1468–1480. ACM/SIAM, Jan. 2010.

[27] D. Micciancio and M. Walter. Fast lattice point enumeration with minimal overhead. In *Proceedings of SODA*, pages 276–294. ACM/SIAM, Jan. 2015.

[28] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of ACM*, 56(6):34, Sept. 2009. Preliminary version in STOC 2005.

[29] C.-P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science*, 53(2–3):201–224, Aug. 1987.

[30] C.-P. Schnorr. Lattice reduction by random sampling and birthday methods. In *STACS*, pages 145–156, 2003.

[31] V. Shoup. Ntl: a library for doing number theory. Available at `http://www.shoup.net/ntl/index.html`.