École Normale Supérieure de Lyon

Réseaux Euclidiens : Algorithmes et Cryptographie

Euclidean Lattices: Algorithms and Cryptography

Damien Stehlé

Chargé de recherche au CNRS

Mémoire d'habilitation à diriger des recherches

présenté le **14 octobre 2011**, après avis des rapporteurs

Arjen LENSTRA, Professor, ÉPFL
Oded REGEV, Directeur de recherche, CNRS
Arne STORJOHANN, Associate professor, University of Waterloo

devant le jury composé de

Karim BELABAS, Professeur, Université de Bordeaux 1
Pascal KOIRAN, Professeur, ÉNS de Lyon
Arjen LENSTRA, Professor, ÉPFL
Bruno SALVY, Directeur de recherche, INRIA
Arne STORJOHANN, Associate professor, University of Waterloo
Brigitte VALLÉE, Directrice de recherche, CNRS
Chee K. YAP, Professor, New York University

Numéro d'ordre xx-yyyy

Exhaustive list of publications

The publications are sorted by themes first, and then in anti-chronological order, regardless of the publication type (book, article, survey, etc).

LLL-type reduction algorithms

- X.-W. Chang, D. Stehlé and G. Villard. *Perturbation Analysis of the QR factor R in the Context of LLL Lattice Basis Reduction*. To appear in Mathematics of Computation.
- A. Novocin, D. Stehlé and G. Villard. *An LLL-reduction algorithm with quasi-linear time complexity*. In the proceedings of STOC 2011.
- I. Morel, D. Stehlé and G. Villard. Analyse numérique et réduction de réseaux. Technique et Science Informatiques, 2010.
- I. Morel, D. Stehlé and G. Villard. *H-LLL: Using Householder inside LLL*. In the proceedings of ISSAC 2009.
- P. Nguyen and D. Stehlé. *An LLL Algorithm with Quadratic Complexity*. SIAM Journal on Computing, 2009.
- D. Stehlé. *Floating-point LLL: theoretical and practical aspects*. Chapter of "The LLL Algorithm, survey and applications", P. Nguyen and B. Vallée (Eds), Springer, 2009.
- A. Akhavi and D. Stehlé. *Speeding-up Lattice Reduction with Random Projections*. In the proceedings of LATIN 2008.
- P. Nguyen and D. Stehlé. *LLL on the Average*. In the proceedings of ANTS-VII, 2006.
- P. Nguyen and D. Stehlé. *Floating-point LLL Revisited*. In the proceedings of EUROCRYPT 2005.

Solving the Shortest and Closest Vector Problems

- G. Hanrot, X. Pujol and D. Stehlé. *Analyzing Blockwise Lattice Algorithms using Dynamical Systems*. To appear in the proceedings of CRYPTO 2011.
- G. Hanrot, X. Pujol and D. Stehlé. *Algorithms for the Shortest and Closest Lattice Vector Problems*. Invited contribution for IWCC 2011.

- J. Detrey, G. Hanrot, X. Pujol and D. Stehlé. *Accelerating Lattice Reduction with FPGAs*. In the proceedings of LATINCRYPT 2010.
- D. Stehlé and X. Pujol. *Solving the Shortest Lattice Vector Problem in Time* $2^{2.465n}$. IACR eprint 2009/605.
- D. Stehlé and X. Pujol. *Rigorous and efficient short lattice vectors enumeration*. In the proceedings of ASIACRYPT 2008.
- G. Hanrot and D. Stehlé. *Worst-Case Hermite-Korkine-Zolotarev Reduced Lattice Bases*. INRIA research report, 2008.
- G. Hanrot and D. Stehlé. *Improved Analysis of Kannan's Shortest Lattice Vector Algorithm*. In the proceedings of CRYPTO 2007.

Other algorithmic aspects of lattices

- C. Fieker and D. Stehlé. *Short Bases of Lattices over Number Fields*. In the proceedings of ANTS-IX, 2010.
- D. Stehlé and M. Watkins. *On the Extremality of an 80-Dimensional Lattice*. In the proceedings of ANTS-IX, 2010.
- P. Nguyen and D. Stehlé. *Low-Dimensional Lattice Basis Reduction Revisited (Full Version)*. Transactions on Algorithms, 2009.
- P. Nguyen and D. Stehlé. *Low-Dimensional Lattice Basis Reduction Revisited (Extended Abstract)*. In the proceedings of ANTS-VI, 2004.

Lattice-based cryptography

- D. Stehlé and R. Steinfeld. *Making NTRU as secure as worst-case problems over ideal lattices*. In the proceedings of EUROCRYPT 2011.
- D. Stehlé and R. Steinfeld. *Faster Fully Homomorphic Encryption*. In the proceedings of ASIACRYPT 2010.
- D. Stehlé, R. Steinfeld, K. Tanaka and K. Xagawa. *Efficient Public-Key Encryption Based on Ideal Lattices*. In the proceedings of ASIACRYPT 2009.

Computer arithmetic

- J.-M. Muller, N. Brisebarre, F. de Dinechin, C.-P. Jeannerod, V. Lefèvre, G. Melquiond, N. Revol, D. Stehlé and S. Torres. *Handbook of Floating-Point Arithmetic*. Birkhauser, 2009.
- G. Hanrot, V. Lefèvre, D. Stehlé and P. Zimmermann. *Worst Cases of a Periodic Function for Large Arguments*. In the proceedings of ARITH'18, 2007.

- V. Lefèvre, D. Stehlé and P. Zimmermann. *Worst Cases for the Exponential Function in the IEEE 754r decimal64 Format*. In a special LNCS volume, following the Dagstuhl seminar number 06021 (Reliable Implementation of Real Number Algorithms: Theory and Practice), 2006.
- D. Stehlé. *On the Randomness of Bits Generated by Sufficiently Smooth Functions.* In the proceedings of ANTS-VII, 2006.
- D. Stehlé and P. Zimmermann. *Gal's Accurate Tables Method Revisited*. In the proceedings of ARITH'17, 2005.
- D. Stehlé, V. Lefèvre and P. Zimmermann. *Searching Worst Cases of a One-Variable Function Using Lattice Reduction*. IEEE Transactions on Computers, 2005.
- D. Stehlé, V. Lefèvre and P. Zimmermann. *Worst Cases and Lattice Reduction*. In the proceedings of ARITH'16, 2003

Misc.

- S. Liu, C. Ling and D. Stehlé. *Decoding by Sampling: A Randomized Lattice Algorithm for Bounded Distance Decoding*. Accepted to IEEE Transactions on Information Theory.
- L. Luzzi, S. Liu, C. Ling and D. Stehlé. *Decoding by Embedding: Correct Decoding Radius and DMT Optimality*. In the proceedings of ISIT 2011.
- D. Stehlé and X.-W. Chang. *Rigorous Perturbation Bounds of Some Matrix Factorizations*. In SIAM Journal on Matrix Analysis and Applications (SIMAX), 2010.
- D. Stehlé and P. Zimmermann. *A Binary Recursive Gcd Algorithm*. In the proceedings of ANTS-VI, 2004.

Contents

Introduction **7**

Notations **11**

1 Reminders on Euclidean Lattices **15**
1.1 Euclidean lattices 15
1.2 Algorithmic problems on lattices 16
1.3 Lattice reduction 17
1.4 Lattice Gaussians 19

2 Computing LLL-Reduced Bases **21**
2.1 A perturbation-friendly definition of LLL-reduction 23
2.2 LLL-reducing using the R-factor of the QR-factorisation 25
2.3 A quasi-linear-time reduction algorithm 27
2.4 Perspectives 30

3 Stronger Lattice Reduction Algorithms **31**
3.1 Cost analysis of the enumeration-based SVP and CVP solvers 33
3.2 Terminating the Schnorr-Euchner BKZ algorithm 36
3.3 Conclusion and perspectives 38

4 Asymptotically Efficient Lattice-Based Encryption Schemes **41**
4.1 A first attempt, from a trapdoor one-way function 44
4.2 A security proof for `NTRUEncrypt` 45
4.3 Perspectives 48

Bibliography **48**

Articles in Appendix

- X.-W. Chang, D. Stehlé and G. Villard. *Perturbation Analysis of the QR factor R in the Context of LLL Lattice Basis Reduction.* To appear in Mathematics of Computation.
- I. Morel, D. Stehlé and G. Villard. *H-LLL: Using Householder inside LLL.* In the proceedings of ISSAC 2009.
- A. Novocin, D. Stehlé and G. Villard. *An LLL-reduction algorithm with quasi-linear time complexity.* In the proceedings of STOC 2011.
- G. Hanrot and D. Stehlé. *Improved Analysis of Kannan's Shortest Lattice Vector Algorithm.* In the proceedings of CRYPTO 2007.
- G. Hanrot, X. Pujol and D. Stehlé. *Analyzing Blockwise Lattice Algorithms using Dynamical Systems.* To appear in the proceedings of CRYPTO 2011.
- D. Stehlé, R. Steinfeld, K. Tanaka and K. Xagawa. *Efficient Public-Key Encryption Based on Ideal Lattices.* In the proceedings of ASIACRYPT 2009.
- D. Stehlé and R. Steinfeld. *Making NTRU as secure as worst-case problems over ideal lattices.* In the proceedings of EUROCRYPT 2011.

Introduction

The present document contains descriptions of results I obtained in the last few years. I chose these specific results because I feel they correspond to the most significant steps towards achieving my main long-term research goals. The purpose of the document is to provide an overview without forcing the reader to delve into the technical proofs of the corresponding articles. The interested reader can however easily access to precisions, as the research articles corresponding to the described results are appended to the text.

My research focuses on devising and analysing faster algorithms for Euclidean lattices and their applications. Lattice algorithms are often classified into two categories: Polynomial-time algorithms for providing interesting representations of lattices, which often means LLL-type algorithms (although Hermite Normal Form algorithms would nicely fit in this category); And slower algorithms that attempt to achieve computationally more demanding tasks. This distinction is clearly artificial (as originally observed by Claus-Peter Schnorr, there exists a whole continuum between the two categories), and tends to become even more so, as ideas developed for one tend to prove useful as well for the other. Nevertheless, the algorithms of the first category deserve specific attention, as they tend to be more practical and have progressively become widespread tools in many fields of computational mathematics and computer science: Amazingly, LLL sometimes seems more famous than the objects it handles! The applications of lattice algorithms are numerous and occur in a very wide variety of fields of mathematics and computer science. The seminal article of Arjen Lenstra, Hendrik Lenstra Jr and László Lovász already considered applications in Computer Algebra (for factoring integer polynomials), Combinatorial Optimisation (for solving Integer Linear Programming instances) and Algorithmic Number Theory (for simultaneous Diophantine approximation). The range of applications of lattices has considerably widened, now including Cryptography (for cryptanalytic purposes, and more recently, for devising cryptographic schemes), Computer Arithmetic, Communications Theory, Computational Group Theory, GPS, etc. For some applications, well-known lattice algorithms can be applied directly, whereas others lead to new mathematical and computational problems on lattices, thus reviving the field.

My PhD thesis was already centred on lattice algorithms and their applications. First, I studied and proposed improvements to lattice reduction algorithms, focusing on strong reductions in tiny dimensions, and on the Lenstra-Lenstra-Lovász reduction in arbitrary dimensions. An important result in that direction was the elaboration, empirical study and implementation of the L^2 algorithm [82, 85, 83, 16]. L^2 was the first algorithm to compute LLL-reduced bases with run-time bounded quadratically with respect to the bit-sizes of the input matrix entries. The algorithmic acceleration was due to the efficient and reliable use of

low-precision floating-point arithmetic to compute (an approximation to) the Gram-Schmidt orthogonalisation of the current lattice basis. This established a link between lattice reduction, traditionally seen as an algebraic procedure, and computer arithmetic and numerical analysis. The second theme of my PhD thesis was the use of lattice reduction to solve difficult problems from the field of computer arithmetic. The main such problem I tackled was the so-called Table's Maker Dilemma: Given a function f over $\mathbb{R}$, an interval I and a precision p_f (e.g., $f = \exp$ on $[1/2, 1)$ with precision $p_f = 53$), compute the minimal sufficient precision p_c such that for any precision p_f floating-point number x in I, the closest precision p_f floating-point number to $f(x)$ can be determined from a precision p_c floating-point approximation to $f(x)$. I proposed a new approach for solving this problem, combining non-linear polynomial approximations to f and Coppersmith's method for finding small roots of bivariate polynomials modulo an integer. The latter itself relies on an LLL-reduction algorithm [117, 116].

After the completion of my PhD thesis, I chose to focus mainly on lattice reduction. I continued investigating numerical analysis techniques for speeding up LLL-reduction algorithms. In particular, with Gilles Villard, we started to progressively replace the Cholesky factorisation used within L^2 for handling the Gram-Schmidt orthogonalisation computations, by the QR-factorisation. These are mathematically equivalent, but the numerical properties of the QR-factorisation are superior, in the sense that smaller precisions may be used while still obtaining meaningful results. Xiao-Wen Chang helped us analysing the sensitivity of the R-factor of the QR-factorisation for LLL-reduced bases, which led to the introduction of a perturbation-friendly modified definition of LLL-reducedness [20]. This study helped us devising an alternative to L^2 relying on Householder's QR-factorisation algorithm [78], and later devising the first LLL-reduction algorithm with quasi-linear complexity with respect to the bit-sizes of the input matrix entries and polynomial complexity with respect to the dimension [88]. Chapter 1 contains the background and reminders necessary for the full document, whereas Chapter 2 is an overview of these results on the LLL-reduction. The reader interested in obtaining more details is referred to the following accompanying articles:

- X.-W. Chang, D. Stehlé and G. Villard. *Perturbation Analysis of the QR factor R in the Context of LLL Lattice Basis Reduction*. To appear in Mathematics of Computation.
- I. Morel, D. Stehlé and G. Villard. *H-LLL: Using Householder inside LLL*. In the proceedings of ISSAC 2009.
- A. Novocin, D. Stehlé and G. Villard. *An LLL-reduction algorithm with quasi-linear time complexity*. In the proceedings of STOC 2011.

Chapter 3 is devoted to algorithms for solving problems on Euclidean lattices that are out of reach of LLL-type algorithms. In 2006, Guillaume Hanrot and I started working on the Kannan-Fincke-Pohst algorithm for solving the Shortest and Closest Lattice Vector Problems. We improved its complexity analysis, and then, together with Xavier Pujol, we studied its numerical and implementation facets [44, 95, 23]. More recently, we investigated the use of a low-dimensional SVP solver for computing bases that are reduced in a stronger sense than LLL's. More specifically, we showed that a slightly simplified version of the Schnorr and Euchner BKZ algorithm [105, 106] may be terminated within a polynomial number of

iterations while still providing bases of excellent quality [43]. The results of this chapter correspond to the following accompanying articles:

- G. Hanrot and D. Stehlé. *Improved Analysis of Kannan's Shortest Lattice Vector Algorithm*. In the proceedings of CRYPTO 2007.
- G. Hanrot, X. Pujol and D. Stehlé. *Analyzing Blockwise Lattice Algorithms using Dynamical Systems*. To appear in the proceedings of CRYPTO 2011.

During my secondment to the University of Sydney and to Macquarie University (between 2008 and 2010), and in collaboration with Ron Steinfeld, I started working in a third field related to the computational aspects of Euclidean lattices. Instead of devising faster algorithms for solving computational problems, the aim was to exploit the apparent computational hardness of some problems on lattices to derive secure cryptographic functions. Lattice-based cryptography started in the mid-90's with Ajtai's seminal worst-case to average-case reduction [3]. It boomed about five years ago, with the elaboration of numerous cryptographic schemes (see [74] for a recent survey). The facet I am most interested in is to use structured lattices corresponding to ideals and modules over rings of integers of some number fields (typically a cyclotomic fields of orders that are powers of 2) to achieve improved efficiency and/or new functionalities. In this vein, together with Ron Steinfeld, Keisuke Tanaka and Keita Xagawa, we proposed the first encryption scheme with quasi-optimal key sizes and encryption/decryption performances, that is provably secure, assuming the exponential quantum worst-case hardness of standard problems on ideal lattices [119]. By building upon recent tools concurrently and independently developed by Lyubashevsky, Peikert and Regev [69], we proved that the famous NTRU encryption scheme [54, 55] can be slightly modified so that it allows for a security proof under a similar assumption [118]. Chapter 4 is devoted to these results.

- D. Stehlé, R. Steinfeld, K. Tanaka and K. Xagawa. *Efficient Public-Key Encryption Based on Ideal Lattices*. In the proceedings of ASIACRYPT 2009.
- D. Stehlé and R. Steinfeld. *Making NTRU as secure as worst-case problems over ideal lattices*. In the proceedings of EUROCRYPT 2011.

Writing this document was an excellent opportunity for me to clarify and put in perspective the results I obtained in the last few years. In particular, it has allowed me to take the time to re-think and structure my research targets. These goals are succinctly overviewed in the "Perspectives" sections of each one of the different chapters. Although lattice algorithms and cryptographic applications will remain my core research area, I intend to broaden my research scope to a larger range of applications of Euclidean lattices, including communications theory (e.g., MIMO technology), numerical analysis (e.g., using lattice algorithms to improve numerical stability), and computational number theory (e.g., units of and modules over the rings of integers of number fields). Looking at the same object from many different angles will hopefully leads to a deeper understanding of its inner workings.

Notations

For a matrix B, we let B^T denote the transpose of B. Furthermore, if B is square, then we will let B^{-T} denote the transpose of its inverse. Also, for any matrix B, the notation $|B|$ will refer to the same matrix where the coefficients have been replaced by their absolute values. The identity matrix will be denoted by I. If $(x_i)_{i \leq n} \in \mathbb{R}^n$, we let $\mathrm{diag}(x_i)$ denote the diagonal matrix whose diagonal coefficients are the x_i's. We let $\mathcal{D}_n$ and $\mathcal{D}_n^+$ respectively denote the sets of n-dimensional diagonal matrices and n-dimensional diagonal matrices with positive diagonal coefficients. The notation $\|B\|_2$ refers to the standard matrix norm induced by the vectorial Euclidean norm.

Vectors will always be denoted by bold-case letters. If two vector $\mathbf{b}$ and $\mathbf{c}$ have matching dimensions, their inner product $\sum_i b_i c_i$ will be denoted by $\langle \mathbf{b}, \mathbf{c} \rangle$. By default, the notation $\|\mathbf{b}\|$ corresponds to the Euclidean norm of $\mathbf{b}$. If $S \subseteq \mathbb{R}^n$, we let $\mathrm{Span}(S)$ denote the vectorial subspace of $\mathbb{R}^n$ spanned by the elements of S. The set of all $n \times n$ matrices over a ring R that are invertible (over R) will be denoted by $\mathrm{GL}_n(R)$. The notation $\mathcal{B}_n(\mathbf{c}, r)$ refers to the n-dimensional (closed) ball of centre $\mathbf{c}$ and radius r.

If S is a finite set, its cardinality is denoted by $|S|$. If S is countable set and f is a function defined over S taking non-negative values, then we let $f(S) \in [0, +\infty]$ denote $\sum_{x \in S} f(x)$.

We use the standard Landau notations $O(\cdot)$, $o(\cdot)$, $\omega(\cdot)$ and $\Omega(\cdot)$. We also use the notations $\widetilde{O}(\cdot)$ and $\widetilde{\Omega}(\cdot)$ fro hiding poly-logarithmic factors. E.g., the function $n \mapsto n^2 \log^c n$ is $\widetilde{O}(n^2)$ for any constant c. The notation $\mathrm{poly}(n)$ denotes any polynomial in n. When a function decreases faster than n^{-c} for any constant $c > 0$, we say it is negligible (or, equivalently, that it is $n^{-\omega(1)}$).

If D is a distribution, the notation $x \hookleftarrow D$ means we sample x with distribution D. If a set S is finite, we let $U(S)$ denote the uniform distribution on S. Also, the probability that an event X occurs will be denoted by $\Pr[X]$. If two distributions D_1 and D_2 are defined over the same support S and if that support is countable, then the statistical distance between D_1 and D_2 is defined as $\Delta(D_1, D_2) = \frac{1}{2} \sum_{x \in S} |D_1(x) - D_2(x)|$.

The notation $\lfloor x \rceil$ denotes an arbitrary integer closest to x. We will use a standard base-2 arbitrary precision floating-point model, such as described in [50, Sec. 2.1]. The notation $\diamond(a)$ refers to the floating-point rounding of a (the working precision being given by the context).

CHAPTER 1

Reminders on Euclidean Lattices

The aim of this chapter is to recall the necessary mathematical background. More in-depth and comprehensive introductions to lattices are available in [41, 115]. Detailed accounts on the computational aspects of lattices include [66, 86, 72, 26, 97].

1.1 Euclidean lattices

A *Euclidean lattice* L a discrete additive subgroup of a Euclidean space. When the latter is $\mathbb{R}^n$, we call n the embedding dimension of the lattice. Equivalently, a lattice in $\mathbb{R}^n$ can be defined as the set of all linear integer combinations of linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_d \in \mathbb{R}^n$, in which case we write:

$$L\left[(\mathbf{b}_i)_{i\leq d}\right] = \left\{\sum_{i\leq d} x_i\mathbf{b}_i : (x_i)_{i\leq d} \in \mathbb{Z}^d\right\} = \sum_{i\leq d} \mathbb{Z}\mathbf{b}_i.$$

We say that the $\mathbf{b}_i$'s form a *basis* of the lattice they span. A lattice may have many bases, but they share the same cardinality d ($\leq n$), which is called the *dimension* of the lattice. The most common way to represent a lattice is to encode it by a basis, i.e., by an $n \times d$ matrix whose columns are the coordinates of the basis vectors. Several situations are of particular interest: When $d = n$, the lattice is said *full-rank*; and when $L \subseteq \mathbb{Z}^n$ (resp. $\mathbb{Q}^n$), the lattice is said integral (resp. rational). For the sake of simplicity, we will restrict ourselves to full-rank lattices, and very often (but not always) to rational lattices.

Unless $d = n \leq 1$, a (full-rank) lattice has infinitely many bases. The bases of a given lattice are obtained from one another by *unimodular* transformations, i.e., invertible integer linear maps. More precisely, if $(\mathbf{b}_i)_{i\leq n}$ is a basis of a lattice L, a tuple $(\mathbf{c}_i)_{i\leq n}$ is also a basis of L if and only if there exists $U \in \mathrm{GL}_n(\mathbb{Z})$ such that $(\mathbf{c}_i)_{i\leq n} = (\mathbf{b}_i)_{i\leq n} \cdot U$. Figure 1.1 gives a two-dimensional lattice with two different bases.

Given a basis of a lattice L, it is of interest to obtain information that is intrinsic to L, i.e., independent of the particular representation of L. The dimension n and embedding dimension n are two such lattice invariants. Popular lattice invariants also include:

- The minimum $\lambda_1(L)$ is the (Euclidean) norm of a shortest non-zero vector of L,
- The successive minima are defined by $\lambda_i(L) = \min(r : \dim \mathrm{Span}(L \cap \mathcal{B}_n(\mathbf{0}, r)) \geq i)$ for all $i \leq n$;
- The determinant $\det(L) = \lim_{r\to\infty} |\mathcal{B}_n(\mathbf{0}, r) \cap L| / \mathrm{vol}(\mathcal{B}_n(\mathbf{0}, r))$ quantifies the density of the lattice in its linear span;

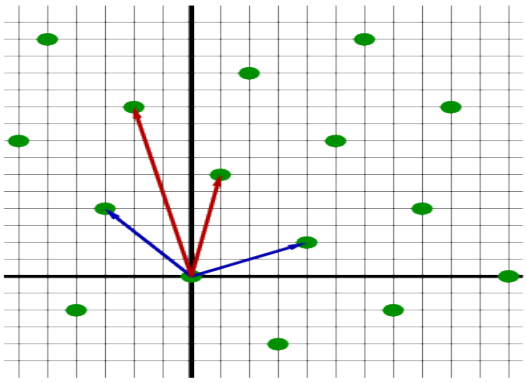

Figure 1.1: A two-dimensional lattice along with two of its bases.

- The covering radius $\rho(L)$ is the largest distance to L of a point in the linear span of L.

Minkowski's theorem provides a link between the minima and the determinant. It states that any lattice L of dimension n satisfies:

$$\prod_{i \leq n} \lambda_i(L) \leq \sqrt{n}^n \cdot \det(L).$$

This implies that the finiteness of the maximum over all n-dimensional lattices L of the quantity $\lambda_1(L)^2 / \det(L)^{2/n}$. This maximum, called *Hermite's constant* in dimension n, will be denoted by γ_n (and we have $\gamma_n \leq n$).

Finally, in order to study a given lattice L, it often proves useful to consider its *dual* lattice $\widehat{L} = \{\mathbf{c} \in \mathrm{Span}(L) : \forall \mathbf{b} \in L, \langle \mathbf{b}, \mathbf{c} \rangle \in \mathbb{Z}\}$. If B is a basis matrix of L, then as the columns of the matrix B^{-T} form a basis of the dual $\widehat{L}$.

1.2 Algorithmic problems on lattices

The most studied algorithmic problems on Euclidean lattices are computational tasks naturally related to the lattice invariants described in the previous section. There exist many variants of the problems we give below, but describing them all is not the purpose of this chapter. We only give those we will consider later on. Also, in order to avoid irrelevant technicalities due to real numbers, the inputs to these problems are restricted to being rational.

SVP_γ. The Shortest Vector Problem with parameter $\gamma \geq 1$ is as follows: Given a basis $(\mathbf{b}_i)_{i \leq n}$ of a rational lattice L, find $\mathbf{b} \in L$ such that $0 < \|\mathbf{b}\| \leq \gamma \cdot \lambda_1(L)$.

SIVP_γ. The Shortest Independent Vectors Problem with parameter $\gamma \geq 1$ is as follows: Given a basis $(\mathbf{b}_i)_{i \leq n}$ of a rational lattice L, find $(\mathbf{c}_i)_{i \leq n} \in L^n$ linearly independent such that $\max_i \|\mathbf{c}_i\| \leq \gamma \cdot \lambda_n(L)$.

HSVP_γ. The Hermite Shortest Vector Problem with parameter $\gamma \geq 1$ is as follows: Given a basis $(\mathbf{b}_i)_{i \leq n}$ of a rational lattice L, find $\mathbf{b} \in L$ such that $0 < \|\mathbf{b}\| \leq \gamma \cdot (\det L)^{1/n}$.

CVP_γ. The Closest Vector Problem with parameter $\gamma \geq 1$ is as follows: Given a basis $(\mathbf{b}_i)_{i \leq n}$ of a rational lattice L and a target $\mathbf{t} \in \mathrm{Span}(L)$, find $\mathbf{b} \in L$ such that $\|\mathbf{b} - \mathbf{t}\| \leq \gamma \cdot \mathrm{dist}(\mathbf{t}, L)$.

BDD_γ. The Bounded Distance Decoding Problem with parameter $\gamma \geq 1$ is as follows: Given a basis $(\mathbf{b}_i)_{i \leq n}$ of a rational lattice L and a target $\mathbf{t} \in \mathrm{Span}(L)$ such that $\mathrm{dist}(\mathbf{t}, L) \leq \gamma^{-1} \cdot \lambda_1(L)$, find $\mathbf{b} \in L$ such that $\|\mathbf{b} - \mathbf{t}\| = \mathrm{dist}(\mathbf{t}, L)$.

Clearly, the complexity of these problems grows with n and decreases with γ. The decisional variant of SVP_γ (deciding whether the minimum of a given lattice is ≤ 1 or $\geq \gamma$, under the promise that we are in one of these situations) is known to be NP-hard under randomised reductions for small values of γ [4, 47]. The same holds for SIVP_γ and CVP_γ under deterministic reductions [28, 24]. Unfortunately, the largest values of γ for which such results are known to hold remain quite small (smaller than n^c for any $c > 0$), but these problems seem to remain very hard to solve even for larger values of γ. The best known algorithms for solving these problems for $\gamma \leq \mathrm{poly}(n)$ all have exponential complexity bounds and are believed to be at least exponential-time in the worst case [77, 76, 44, 96] and the survey [42]. Schnorr's algorithm [104] using [76] as a subroutine allows one to trade cost for output quality. It is the best known algorithm for intermediate values of γ, reaching $\gamma = k^{O(n/k)}$ in time and space $\mathrm{poly}(n) \cdot 2^{O(k)}$ (up to a factor that is polynomial in the bit-size of the input). By choosing $k = O(\log n)$, one obtains a polynomial-time algorithm for $\gamma = 2^{O(n \frac{\log \log n}{\log n})}$. Beating the trade-off achieved by Schnorr's hierarchy is a long-standing open problem.

It is also worth noting at this stage that it is not currently known how to exploit quantum computing to outperform classical algorithms for solving these problems. However, no argument is known either for discrediting such a possibility.

1.3 Lattice reduction

Lattice reduction is a representation paradigm. Given a basis of a lattice, the aim is to find another basis of the same lattice with guaranteed norm and orthogonality properties. All the known algorithms for solving the problems mentioned in the previous section rely at least at some stage, or completely, on lattice reduction. Note that the word *reduction* is ambiguous, as it can equally refer to the state of being reduced, or to the process of reducing. However, the meaning is usually clear from the context.

In order to be able to properly define several notions of reduction, we first recall some facts on the QR matrix factorisation and its relationship to the Gram-Schmidt orthogonalisation.

Any full column rank matrix $B \in \mathbb{R}^{n \times n}$ (which can be seen as the basis matrix of a lattice) can be factored as $B = QR$ where $Q \in \mathbb{R}^{n \times n}$ is an orthogonal matrix (i.e., $Q \cdot Q^T = Q^T \cdot Q = I$), and $R \in \mathbb{R}^{n \times n}$ is upper triangular with positive diagonal coefficients. Note that the R-factor of B can also be obtained from the *Cholesky factorisation* $G = R^T R$ of the positive definite matrix $G = B^T B$, called the *Gram matrix* of B. The *QR matrix factorisation* encodes the same information as the Gram-Schmidt orthogonalisation (GSO for short): the former lends itself more easily to algebraic and numeric techniques, while the latter conveys more geometrical intuition. The *Gram-Schmidt orthogonalisation* of a basis $(\mathbf{b}_i)_{i \leq n}$ is the orthogonal family $(\mathbf{b}_i^*)_{i \leq n}$ where $\mathbf{b}_i^*$ is the projection of $\mathbf{b}_i$ orthogonally to the span of $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$. More explicitly

$$\mathbf{b}_i^* = \mathbf{b}_i - \sum_{j<i} \mu_{ij} \mathbf{b}_j^* \text{ with } \mu_{ij} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\|\mathbf{b}_j^*\|^2}.$$

If $B = (\mathbf{b}_i)_{i\leq n}$ has QR-factorisation $B = QR$ and GSO $(\mathbf{b}_i^*)_{i\leq n}$, then for $i \leq n$ the ith column of Q is $\frac{\mathbf{b}_i^*}{\|\mathbf{b}_i^*\|}$, and for $1 \leq i \leq j \leq n$ we have $r_{ij} = \mu_{ji}\|\mathbf{b}_i^*\| = \mu_{ji}r_{ii}$.

The QR-factorisation and GSO provide informations on the lattice invariants. If $(\mathbf{b}_i)_{i\leq n}$ is a basis of a lattice L, then we have:

$$\begin{aligned}
\lambda_i(L) &\geq \min_{j\geq i} \|\mathbf{b}_j^*\| \text{ for all } i \leq n,\\
\lambda_i(L) &\leq \max_{j\leq i} \|\mathbf{b}_j\| \text{ for all } i \leq n,\\
\det(L) &= \prod_{i\leq n} \|\mathbf{b}_i^*\|,\\
\rho(L) &\leq \frac{1}{2}\sqrt{\sum_{i\leq n} \|\mathbf{b}_i^*\|^2}.
\end{aligned}$$

We say that a basis $(\mathbf{b}_i)_{i\leq n}$ is *size-reduced* if $|\mu_{ij}| \leq 1/2$ (or, equivalently, if $|r_{ji}| \leq r_{jj}/2$) for all $i > j$. Other definitions of size-reducedness have been introduced, with computational advantages over this classical definition, but we postpone this discussion to Chapter 2. The basis $(\mathbf{b}_i)_{i\leq n}$ is said *Lenstra-Lenstra-Lovász*-reduced with parameter $\delta \in (1/4, 1]$ (δ-LLL-reduced for short) if it is size-reduced and for all $i < d$ we have $\delta r_{ii}^2 \leq r_{i+1i+1}^2 + r_{ii+1}^2$ (or, equivalently, $\delta\|\mathbf{b}_i^*\|^2 \leq \|\mathbf{b}_{i+1}^* + \mu_{i+1i}\mathbf{b}_i^*\|^2$). The latter condition, often ascribed to Lovász, states that once projected orthogonally to $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$, the $i+1$th vector is almost longer than the ith vector. Figure 2.1 illustrates this definition in dimension 2.

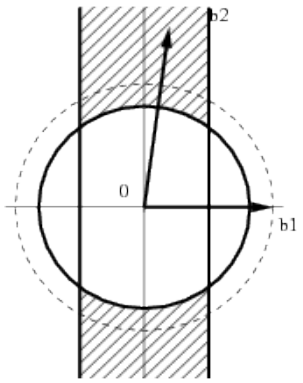

Figure 1.2: The hashed area is the set of possible locations for $(\mathbf{b}_1, \mathbf{b}_2)$ to be δ-LLL-reduced.

LLL-reduction has the twofold advantage of being computable in polynomial-time (using the LLL algorithm [65]) and providing bases of quite decent quality. Among others, an LLL-reduced basis $(\mathbf{b}_i)_{i\leq n}$ of a lattice L satisfies the following properties (with $\alpha =$

$(\delta - 1/4)^{-1} \geq \sqrt{3}/2$):

$$\begin{aligned} r_{ii} &\leq \alpha \cdot r_{i+1i+1} \text{ for all } i < n, \\ \|\mathbf{b}_i\| &\leq \alpha^{i-1} \cdot r_{ii} \text{ for all } i \leq n, \\ \alpha^{i-d} \cdot r_{ii} &\leq \lambda_i(L) \leq \alpha^i \cdot r_{ii} \text{ for all } i \leq n, \\ \|\mathbf{b}_1\| &\leq \alpha^{\frac{n-1}{2}} \cdot (\det(L))^{\frac{1}{n}}, \\ \prod_{j \leq n} \|\mathbf{b}_j\| &\leq \alpha^{\frac{n(n-1)}{2}} \cdot \det(L). \end{aligned}$$

Oppositely, the quality of *Hermite-Korkine-Zolotarev-reduced* bases (HKZ-reduced for short) is much higher, but computing an HKZ-reduced basis of a lattice L from an arbitrary basis of L is polynomial-time equivalent to solving SVP_γ for $\gamma = 1$. A basis $(\mathbf{b}_i)_{i \leq n}$ is said HKZ-reduced if it is size-reduced and if for any $i \leq n$, we have $\|\mathbf{b}_i^*\| = \lambda_1(L[(\mathbf{b}_j^{(i)})_{j \geq i}])$, where $\mathbf{b}_j^{(i)} = \mathbf{b}_j - \sum_{k<i} \mu_{jk} \mathbf{b}_k^*$ is the projection of the vector $\mathbf{b}_j$ orthogonally to $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$. As a direct consequence of Minkowski's theorem, we have:

$$\forall i \leq n,\ \|\mathbf{b}_i^*\| \leq \sqrt{n-i+1} \left(\prod_{j=i}^{n} \|\mathbf{b}_j^*\| \right)^{\frac{1}{n-i+1}}.$$

In 1987, Schnorr introduced a hierarchy of reductions ranging from LLL to HKZ [103]. All known algorithms mentioned in the previous section for solving the four mentioned problems for intermediate values of γ attempt to achieve Schnorr's Block-Korkine-Zolotarev reduction (BKZ for short) or variants thereof (see, e.g., [103, 105, 106, 33, 34]). A basis $(\mathbf{b}_i)_{i \leq n}$ is said BKZ_β-reduced for $\beta \in [2, n]$ if it is size-reduced and if for all $i \leq n$ the vectors $\mathbf{b}_i^*, \mathbf{b}_{i+1}^{(i)}, \ldots, \mathbf{b}_{\min(i+\beta-1,n)}^{(i)}$ form an HKZ-reduced basis (in dimension $\min(n-i+1, \beta)$).

1.4 Lattice Gaussians

Discrete Gaussian distributions with lattice supports have recently arisen as a powerful tool in lattice-based cryptography. They have been first used by Micciancio and Regev [73] to improve on Ajtai's worst-case to average-case reduction [3]. Another major breakthrough occurred in 2008, when Gentry, Peikert and Vaikutanathan [39] showed that Klein's algorithm [61] may be used to sample points according to these distributions (or, more precisely, from distributions whose statistical distances to desired discrete Gaussians is small).

Let $L \subseteq \mathbb{R}^n$ be a full-rank lattice. The discrete Gaussian distribution $D_{L,\sigma,\mathbf{c}}$ of support L, centre $\mathbf{c} \in \mathbb{R}^n$ and standard deviation σ is defined by:

$$\forall \mathbf{x} \in L :\ D_{L,\sigma,\mathbf{c}}(\mathbf{x}) = \frac{\rho_{\mathbf{c},\sigma}(\mathbf{x})}{\sum_{\mathbf{b} \in L} \rho_{\mathbf{c},\sigma}(\mathbf{b})},$$

where $\rho_{\mathbf{c},\sigma}(\mathbf{x}) = \exp(-\pi \|\mathbf{x} - \mathbf{b}\|^2 / \sigma^2)$. The subscripts $\mathbf{c}$ and L will be omitted when $\mathbf{c} = \mathbf{0}$ and $L = \mathbb{Z}^n$ respectively. Two Gaussian distributions with centre $\mathbf{0}$ and support $\mathbb{Z}^2$ but different standard deviations are presented in Figure 1.4.

As can be observed, the larger the standard deviation, the smoother the distribution looks. In fact, the larger the standard deviation, the closer the behaviour of the discrete

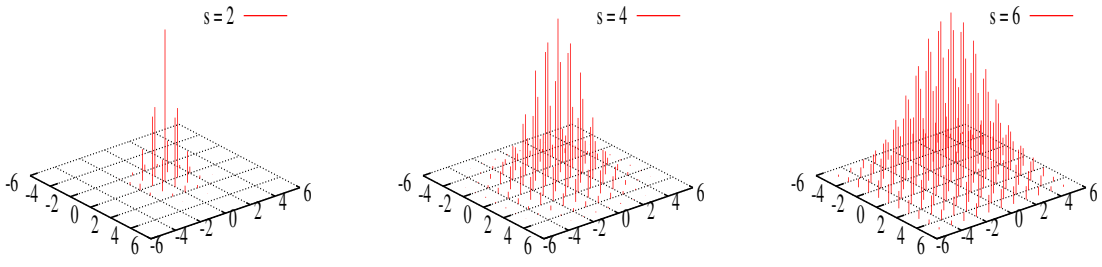

Figure 1.3: Three discrete Gaussian distributions with support $\mathbb{Z}^2$ and centre $\mathbf{0}$, but different standard deviations s.

Gaussian to that of a continuous Gaussian. This phenomenon is quantified by the so-called *smoothing parameter*. For a lattice L and a parameter $\varepsilon > 0$, the ε-smoothing parameter of L is defined by $\eta_\varepsilon(L) = \min(\sigma : \rho_{\mathbf{0},1/\sigma}(\widehat{L} \setminus \mathbf{0}) \leq \varepsilon)$. For any $\varepsilon \in (0,1)$, we have (see [73, 91]):

$$\eta_\varepsilon(L) \leq \sqrt{\frac{\ln(2n + 1/\varepsilon))}{\pi}} \cdot \min\left(\lambda_n(L), \frac{1}{\lambda_1^\infty(\widehat{L})}\right),$$

where $\lambda_1^\infty(\widehat{L})$ stands for the first minimum of the dual $\widehat{L}$ with respect to the infinity norm.

We will use the following properties of lattice Gaussians (proved in [39, 73]):

- For any full-rank lattice $L \subseteq \mathbb{R}^n$, $\mathbf{c} \in \mathbb{R}^n$, $\varepsilon \in (0, 1/3)$ and $\sigma \geq \eta_\varepsilon(L)$, we have $\Pr_{\mathbf{b} \hookleftarrow D_{L,\sigma,\mathbf{c}}}[\|\mathbf{b}\| \geq \sigma\sqrt{n}] \leq 2^{-n+1}$.
- For any full-rank lattices $L' \subseteq L \subseteq \mathbb{R}^n$, $\mathbf{c} \in \mathbb{R}^n$, $\varepsilon \in (0, 1/2)$ and $\sigma \geq \eta_\varepsilon(L')$, we have $\Delta(D_{L,\sigma,\mathbf{c}} \bmod L'; U(L/L')) \leq 2\varepsilon$.

Finally, as we mentioned above, any Gaussian distribution with support a full-rank lattice $L \subseteq \mathbb{Q}^n$ may be sampled from efficiently using a basis $(\mathbf{b}_i)_{i \leq n}$ of L, provided that the desired standard deviation is sufficiently large.

Theorem 1 ([39, Th. 4.1]) *There exists a polynomial-time algorithm that takes as input any basis $(\mathbf{b}_i)_{i \leq n}$ of any lattice $L \subseteq \mathbb{Q}^n$, any centre $\mathbf{c} \in \mathbb{Q}^n$ and any $\sigma = \omega(\sqrt{\log n}) \cdot \max \|\mathbf{b}_i\|$ (resp. $\sigma = \Omega(\sqrt{n}) \cdot \max \|\mathbf{b}_i\|$), and returns samples from a distribution whose statistical distance to $D_{L,\sigma,\mathbf{c}}$ is negligible (resp. exponentially small) with respect to n.*

CHAPTER 2

Computing LLL-Reduced Bases

In their seminal article [65], Lenstra, Lenstra and Lovász both introduced the notion of LLL-reducedness (recalled in Chapter 1), and an algorithm for computing LLL-reduced bases. This algorithm, commonly referred to as LLL or L^3, is recalled in Figure 2.1.

Input: A basis $(\mathbf{b}_i)_{i\leq n}$ of $L \subseteq \mathbb{Z}^n$ and $\delta \in (1/4, 1)$.
Output: A δ-LLL-reduced basis.
1. Compute the rational GSO, i.e., all the $\mu_{i,j}$'s and $\mathbf{b}_i^*$'s.
2. $\kappa := 2$. While $\kappa \leq n$ do
3. Size-reduce the vector $\mathbf{b}_\kappa$ using the size-reduction algorithm of Figure 2.2.
4. If $\delta \cdot \|\mathbf{b}_{\kappa-1}^*\|^2 \leq \|\mathbf{b}_\kappa^*\|^2 + \mu_{\kappa\kappa-1}^2\|\mathbf{b}_{\kappa-1}^*\|^2$, then set $\kappa := \kappa + 1$.
5. Else swap $\mathbf{b}_{\kappa-1}$ and $\mathbf{b}_\kappa$, update the GSO and set $\kappa := \max(2, \kappa - 1)$.
6. Output $(\mathbf{b}_i)_{i\leq n}$.

Figure 2.1: The L^3 algorithm.

Input: A basis $(\mathbf{b}_i)_{i\leq n}$ of $L \subseteq \mathbb{Z}^n$, its GSO and an index κ.
Output: The same basis but with the vector $\mathbf{b}_\kappa$ size-reduced, and the updated GSO.
1. For $i = \kappa - 1$ down to 1 do
2. $\mathbf{b}_\kappa := \mathbf{b}_\kappa - \lceil \mu_{\kappa,i} \rfloor \cdot \mathbf{b}_i$.
3. Update the GSO accordingly.

Figure 2.2: The size-reduction algorithm.

In this chapter, we will use the variable $\beta = \max_i \log \|\mathbf{b}_i\|$, using the input $\mathbf{b}_i$'s. The costs of LLL and its variants will be bounded with respect to both n and β.

The LLL algorithm is polynomial-time but remains quite slow. Its inefficiency stems from the following combination of drawbacks:

- The GSO computations are performed in *exact* rational arithmetic, with numerators and denominators of possibly huge bit-sizes $O(n\beta)$.
- The basis computations are performed in *exact* integer arithmetic. The involved integers have smaller bit-sizes $O(n + \beta)$ than the rationals involved in the GSO computations, but still significantly contribute to the cost, as there are up to $O(n^2\beta)$ loop iterations (from Steps 3 to 6 of the algorithm of Figure 2.1).

- Finally, many of the size-reduction steps are superfluous. Assume the index κ remains in some small interval $[i_1, i_2]$ during some consecutive loop iterations, then for each iteration LLL performs a full size-reduction of the current vector with respect to all the previous basis vectors (i.e., in the algorithm of Figure 2.2, the index i goes from κ all the way down to 1 every time). But only the GSO quantities $\|\mathbf{b}_i^*\|^2$ and μ_{ij} for $i, j \in [i_1, i_2]$ are useful for correctly deciding the Lovász tests (Step 4 of the algorithm of Figure 2.1).

The first two sources of inefficiency are of an arithmetic flavour, while the third is related to fast linear algebra techniques (subdividing matrices into blocks and using fast matrix multiplication). In this chapter, we will be concerned with the arithmetic aspects of LLL and we will not elaborate on how to save size-reduction operations (see [109, 102, 121] for works in that direction). Table 2.1 summarises the algorithmic improvements for computing LLL-reduced bases over the original LLL algorithm, that are of an arithmetic nature. For the derivation of the bit-complexity upper bounds, we assume fast integer multiplication is used [111, 32]: Two ℓ-bit long integers may be multiplied in time $O(\ell^{1+\varepsilon})$, for some ε that is $o(1)$. Also, it is worth noting that among the described algorithms, only those from [65] and [58] return bases that are genuinely LLL-reduced. The others return bases that are reduced in a sense that is slightly weaker than the LLL-reduction (see Section 2.1 below).

Table 2.1: Bit-complexities of selected LLL-reduction algorithms.

	Bit-complexity	Output reducedness
[65], LLL/L^3	$O(n^{5+\varepsilon}\beta^{2+\varepsilon})$	δ-LLL-reduced
[58]	$O(n^5\beta^2(n+\beta)^{\varepsilon})$	δ-LLL-reduced
[104]	$O(n^4\beta(n+\beta)^{1+\varepsilon})$	(δ,η)-LLL-reduced
[82, 84], L^2	$O(n^{4+\varepsilon}\beta(n+\beta))$	(δ,η)-LLL-reduced
[78] and Section 2.2, H-LLL	$O(n^{4+\varepsilon}\beta(n+\beta))$	(δ,η,θ)-LLL-reduced
[88] and Section 2.3, $\widetilde{\text{L}}^1$	$O(n^{5+\varepsilon}\beta + n^{4+\varepsilon}\beta^{1+\varepsilon})$	(δ,η,θ)-LLL-reduced

The L^2 algorithm was the first to achieve a complexity bound that is quadratic with respect to β. It relies on exact integer operations for the basis matrix computations and on approximate floating-point arithmetic for the underlying GSO computations. By relying on an exact Gram matrix computation (the Gram matrix of the basis $(\mathbf{b}_i)_{i\leq n}$ is the positive symmetric definite matrix $(\langle \mathbf{b}_i, \mathbf{b}_j \rangle)_{i,j\leq n}$) and on the Cholesky factorisation algorithm, the computed approximations of the GSO coefficients can be proven to be close to the genuine GSO coefficients, and the decisions taken by the tests of the LLL algorithm using these approximate data thus remain sufficiently meaningful for making progress during the execution. The cost improvement of L^2 stems from the fact that a low precision of $O(n)$ bits suffices for being able to guarantee correctness: This itself originates from the facts that at any loop iteration the vector $\mathbf{b}_\kappa$ under scope is always such that $(\mathbf{b}_i)_{i<\kappa}$ is a reduced basis and that reduced bases are well-conditioned, guaranteeing that a low precision suffices to obtain meaningful results.

An important drawback of L^2 is its reliance on the Cholesky factorisation algorithm: First, it leads L^2 to require the computation and update of the (exact) Gram matrix; And

second the Cholesky factorisation is much more sensitive to perturbations than the QR-factorisation, leading to requiring higher precisions than a priori necessary. In Sections 2.1 and 2.2 explain how the Cholesky factorisation may be replaced by the QR-factorisation. Section 2.3 presents another step towards improving LLL-reduction algorithms: approximate computations may also be performed on the basis matrices themselves.

2.1 A perturbation-friendly definition of LLL-reduction

The following examples in dimension 2 show that the classical notion of LLL-reduction is not preserved under roundings of the basis vectors. Assume we round each entry of the following matrices at t_1 bits of precision:

$$B_1 := \begin{bmatrix} 1 & 2^{t_1+t_2+1}+2^{t_2} \\ -1 & 2^{t_1+t_2+1} \end{bmatrix} \text{ and } B_2 := \begin{bmatrix} 1 & 2^{t_1}+2^{-1}+2^{-2t_2} \\ 2^{-t_2} & -2^{t_1+t_2} \end{bmatrix}.$$

Then we obtain:

$$\overline{B}_1 := \begin{bmatrix} 1 & 2^{t_1+t_2+1} \\ -1 & 2^{t_1+t_2+1} \end{bmatrix} \text{ and } \overline{B}_2 := \begin{bmatrix} 1 & 2^{t_1}+1 \\ 2^{-t_2} & -2^{t_1+t_2} \end{bmatrix}.$$

The basis matrix B_1 is not reduced as the inner product of the two columns is 2^{t_2}, which can be set arbitrarily large compared to the norm of the first column, by letting t_2 grow to infinity. However, its approximation $\overline{B}_1$ is always reduced, as its columns are orthogonal. Oppositely, the basis matrix B_2 is reduced as soon as $t_2 \geq 1$, while its approximation $\overline{B}_2$ is not reduced.

This phenomenon is unfortunate: It would be convenient (and more efficient!) to be able to decide reducedness by looking only at the most significant bits of the entries of the matrix under scope. But the above examples show that LLL-reducedness is not preserved under roundings, or, more generally, perturbations.

As the definition of LLL-reduction expresses itself in terms of the QR matrix factorisation, it is natural to analyse the sensitivity of the R-factor of an LLL-reduced basis under perturbations. This is a classical topic in numerical analysis [123, 19, 18], but we needed stronger results for our purposes.

Theorem 2 ([20]) *Let $B \in \mathbb{R}^{n\times n}$ be of full column rank with QR factorisation $B = QR$. Let the perturbation matrix $\Delta B \in \mathbb{R}^{n\times n}$ satisfy $\max_i \frac{\|\Delta \mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq \varepsilon$. If*

$$\operatorname{cond}(R)\cdot\varepsilon < \frac{\sqrt{3/2}-1}{n^{3/2}} \quad \textit{with} \quad \operatorname{cond}(R) = \||R||R^{-1}|\|,$$

then $B+\Delta B$ has a unique QR factorisation $B+\Delta B = (Q+\Delta Q)(R+\Delta R)$, and

$$\max_i \frac{\|\Delta \mathbf{r}_i\|}{\|\mathbf{r}_i\|} \leq (\sqrt{6}+\sqrt{3})n^{3/2}\chi(B)\,\varepsilon,$$

where (with $\zeta_{\operatorname{diag}(\delta_i)} := \sqrt{1+\max_{i<j}(\delta_j/\delta_i)^2}$):

$$\chi(B) = \inf_{D\in\mathcal{D}_n^+} \frac{\zeta_D \left\||R||R^{-1}|D\right\|_2 \left\|D^{-1}R\right\|_2}{\|R\|_2}.$$

Given this columnwise perturbation bound, the aim is then to find a variant of the definition of LLL-reduction that is preserved under columnwise perturbations. This is provided by the following definition (a variant of that definition was implicit in [102]).

Definition 1 ([20, Def. 5.3]) *Let $\Xi = (\delta, \eta, \theta)$ with $\eta \in (1/2, 1)$, $\theta > 0$ and $\delta \in (\eta^2, 1)$. Let $B \in \mathbb{R}^{d \times d}$ be non-singular with QR factorisation $B = QR$. The matrix B is Ξ-LLL-reduced if:*

- *For all $i < j$, we have $|r_{ij}| \leq \eta r_{ii} + \theta r_{jj}$;*
- *For all i, we have $\delta \cdot r_{ii}^2 \leq r_{ii+1}^2 + r_{i+1i+1}^2$.*

Let $\Xi_i = (\delta_i, \eta_i, \theta_i)$ be valid LLL-parameters for $i \in \{1, 2\}$. We say that Ξ_1 is stronger than Ξ_2 and write $\Xi_1 > \Xi_2$ if $\delta_1 > \delta_2$, $\eta_1 < \eta_2$ and $\theta_1 < \theta_2$.

Note that for $\theta = 0$, we recover the (δ, η)-LLL-reduction from [82] (which was already implicit in [104]), and that for $(\eta, \theta) = (1/2, 0)$, we recover the classical δ-LLL-reduction. Figure 2.1 illustrates these different types of reduction.

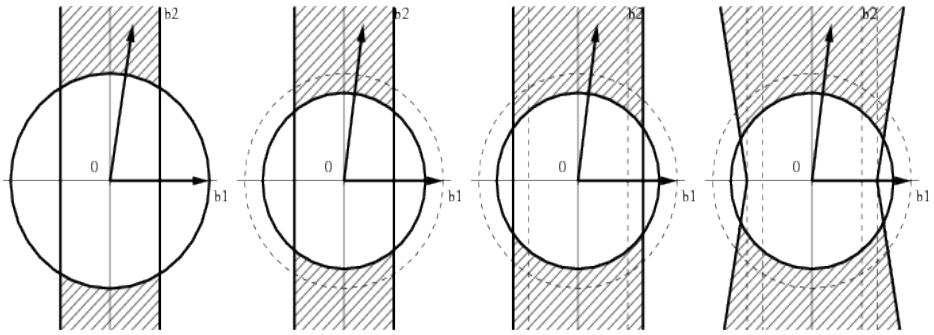

Figure 2.3: The hashed area is the set of vectors $\mathbf{b}_2$ such that $(\mathbf{b}_1, \mathbf{b}_2)$ is (from left to right) $(1, 0, 0)$-LLL, $(\delta, 0, 0)$-LLL, $(\delta, \eta, 0)$-LLL and (δ, η, θ)-LLL.

Note that the Ξ-LLL-reduction and classical δ-LLL-reduction mostly differ when the r_{ii}'s increase, which is the case of the two-dimensional examples above. Also, the quality properties satisfied by δ-LLL-reduced bases (see Section 1.3) are also satisfied by (δ, η, θ)-reduced bases, after replacing $\alpha = (\delta - 1/4)^{-1} \geq \sqrt{3}/2$ by $\alpha = \frac{\theta\eta + \sqrt{(1+\theta^2)\delta - \eta^2}}{\delta - \eta^2}$. Additionally, any (δ, η, θ)-reduced basis B with R-factor R satisfies $\mathrm{cond}(R) \leq \frac{|1-\eta-\theta|\alpha+1}{(1+\eta+\theta)\alpha-1}(1+\eta+\theta)^n \alpha^n = 2^{O(n)}$, allowing us to use Theorem 2.

Finally, the following result, derived from Theorem 2 and the good orthogonality properties of Ξ-reduced bases, shows that the modified notion of LLL-reduction is preserved under column-wise perturbations.

Theorem 3 ([20, Co. 5.1]) *Let $\Xi_1 > \Xi_2$ be valid reduction parameters. There exists a constant c such that for any Ξ_1-LLL-reduced $B \in \mathbb{R}^{n \times n}$ and any $\Delta B \in \mathbb{R}^{n \times n}$ with $\max \frac{\|\Delta \mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-c \cdot n}$, the matrix $B + \Delta B$ is non-singular and Ξ_2-LLL-reduced.*

2.2 LLL-reducing using the R-factor of the QR-factorisation

By combining the above sensitivity analysis of the R-factor under columnwise perturbations, with the backward stability of the Householder QR-factorisation algorithm (see [50, Ch. 19] and [20, Se. 6]), we obtain that if a basis is Ξ-LLL-reduced, then the matrix $\overline{R}$ computed by Householder's algorithm with precision p floating-point arithmetic is a good approximation to the genuine R-factor. Note that any other algorithm computing the R-factor could be equally used, as long as it satisfies a column-wise backward error stability bound such as the one below (up to any multiplicative factor that is polynomial in n): This includes the Givens algorithm based on Givens rotations, and the Modified Gram-Schmidt algorithm [50, Ch. 19].

Theorem 4 *Let $\overline{R}$ be the computed R-factor of the QR factorisation of a given matrix $B \in \mathbb{R}^{n\times n}$ by the Householder algorithm, with precision p floating-point arithmetic. If $80n^2 \cdot 2^{-p} \leq 1$, then there exists an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ such that*

$$B + \Delta B = Q\overline{R} \text{ and } \max \frac{\|\Delta \mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 80n^2 \cdot 2^{-p}.$$

Inputs: A basis $B = (\mathbf{b}_i)_{i\leq n}$ of $L \subseteq \mathbb{Z}^{n\times n}$; a precision p; $\diamond(2^{-cn})$ (for an arbitrary $c > 0$); and a floating-point number $\overline{\delta}$.
Output: A basis of L.
1. Compute an approximation $\overline{\mathbf{r}}_1$ of the first column of the R-factor of B, using Householder's algorithm in precision p.
2. $\kappa := 2$. While $\kappa \leq n$, do
3. Call the algorithm of Figure 2.5 on input $\left[(\mathbf{b}_i)_{i\leq n}, (\overline{\mathbf{r}}_i)_{i<\kappa}, \diamond(2^{-cd}), p\right]$.
4. $s := \diamond(\| \diamond (\mathbf{b}_\kappa)\|^2)$; $s := \diamond(s - \sum_{i\leq\kappa-2} \overline{r}_{i\kappa}^2)$.
5. If $\diamond(\overline{\delta} \cdot \diamond(\overline{r}^2_{\kappa-1\kappa-1})) \leq s$, then $\kappa := \kappa + 1$.
6. Else swap $\mathbf{b}_{\kappa-1}$ and $\mathbf{b}_\kappa$; and set $\kappa := \max(\kappa - 1, 2)$.
7. Return $(\mathbf{b}_i)_{i\leq n}$.

Figure 2.4: The H-LLL algorithm.

The H-LLL algorithm, given in Figure 2.4, mimics the LLL algorithm except that it relies on an approximate R-factor computed and updated using the (floating-point) Householder QR-factorisation algorithm. The operations performed on the exact data (the lattice basis) are derived from approximate values. The fact that these are good approximations to the genuine values allow us to show that H-LLL is correct: It returns Ξ-reduced bases.

Theorem 5 ([78]) *Given as inputs a basis $(\mathbf{b}_i)_{i\leq n}$ of a lattice $L \subseteq \mathbb{Z}^n$, a precision $p = \Theta(n)$, and floating-point numbers $\overline{\delta} \in (1/2, 1)$ and $\diamond(2^{-cn})$, the H-LLL algorithm returns a (δ, η, θ)-LLL-reduced basis $(\mathbf{c}_i)_{i\leq n}$ of L, with δ, η, θ close to $\overline{\delta}$, $1/2 + \diamond(2^{-cn})$ and $\diamond(2^{-cn})$ respectively. Furthermore, its bit-complexity is bounded by*

$$O\left[\left(n + \log \prod \frac{d_i^{\mathbf{b}}}{d_i^{\mathbf{c}}} + \frac{1}{n} \log \prod \frac{\|\mathbf{b}_i\|}{\|\mathbf{c}_i\|}\right) n^{2+\varepsilon}(n + \beta)\right],$$

Inputs: A basis $(\mathbf{b}_i)_{i\leq n}$ of $L \subseteq \mathbb{Z}^{n\times n}$; a precision p; approximations $(\bar{\mathbf{r}}_i)_{i<\kappa}$ of the $\kappa - 1$ first columns of the R-factor of B; $\diamond(2^{-cn})$ (for an arbitrary $c > 0$); a precision p.
Output: A basis $(\mathbf{b}_i)_{i\leq n}$ of L, approximations $(\bar{\mathbf{r}}_i)_{i\leq\kappa}$ of the κ first columns of the R-factor of B.
1. Do
2. Compute $\bar{\mathbf{r}}_\kappa$ using Householder's algorithm (in precision p).
3. For i from $\kappa - 1$ to 1, do
4. $X_i := \lfloor \diamond(\bar{r}_{i\kappa}/\bar{r}_{ii}) \rceil$.
5. For j from 1 to $i - 1$, do $\bar{r}_{j\kappa} := \diamond\left(\bar{r}_{j\kappa} - \diamond\left(X_i \bar{r}_{ji}\right)\right)$.
6. $t := \diamond(\|\mathbf{b}_\kappa\|^2)$; $\mathbf{b}_\kappa := \mathbf{b}_\kappa - \sum_{i<\kappa} X_i \mathbf{b}_i$.
7. Until $\diamond(\|\mathbf{b}_\kappa\|^2) > \diamond(\diamond(2^{-cd}) \cdot t)$.
8. Compute $\bar{\mathbf{r}}_\kappa$ using Householder's algorithm (in precision p).
9. Return $(\mathbf{b}_i)_{i\leq n}$ and $(\bar{\mathbf{r}}_i)_{i\leq\kappa}$.

Figure 2.5: The size-reduction algorithm of H-LLL.

where $\beta = \max_i \log \|\mathbf{b}_i\|$, $\varepsilon = o(1)$ and $d_i^{\mathbf{b}}$ (resp. $d_i^{\mathbf{c}}$) is the determinant of the lattice spanned by the first i columns of the input (resp. output) basis. The complexity bound above is itself $O(n^{4+\varepsilon}\beta(n+\beta))$.

Precise conditions on p, $\overline{\delta} \in (1/2, 1)$, $\diamond(2^{-cn})$, and (δ, η, θ) may be found in [78]. H-LLL has three advantages over L^2. First, it does not require to compute and update the Gram matrix of the current basis. Second, its precision requirement is lower: in the case of (δ, η, θ) close to $(1, 1/2, 0)$, the precision required for ensuring correctness of L^2 tends to $\lceil \log_2 3 + o(1) \rceil \cdot n \lesssim 1.6 \cdot n$, while that of H-LLL tends to n. This is not only an artifact of the worst-case analysis, as it can be observed on actual examples that the numerical performance of H-LLL is superior to that of L^2 (e.g., using the input bases from `http://perso.ens-lyon.fr/damien.stehle/L2.html`. It actually seems that the worst-case bound on the precision required by H-LLL might not be sharp: Checking the reducedness of an LLL-reduced basis can require as low as $\lceil \frac{1}{2} \log_2 3 + o(1) \rceil \cdot n \lesssim 0.8 \cdot n$ precision, but for the moment we do not manage to prove correctness of the size-reduction process with that low a precision. These two facts, on the Gram matrix and the working precision, lead to constant factor improvements. The third advantage of H-LLL over L^2 is its simplified complexity analysis. The analysis of L^2 from [82, 84] required a rather involved amortised analysis for summing the cost bounds for the successive size-reductions. In H-LLL, the corresponding analysis is much simpler, as the cost of a size-reduction is bounded by

$$O\left(n^{1+\varepsilon}(n+\beta)\left(n + \frac{\log \|\mathbf{b}_\kappa^b\|}{\log \|\mathbf{b}_\kappa^e\|}\right)\right),$$

where $\mathbf{b}_\kappa^b$ and $\mathbf{b}_\kappa^e$ denote $\mathbf{b}_\kappa$ before and after the call to the size-reduction algorithm, respectively. Summing such quantities over the successive loop iterations is straightforward. This simplification is not simply a technical stroke of luck: The H-LLL algorithm is vectorial in nature, as all operations are vector operations, and it is no surprise that the cost bound for the size-reduction directly involves the bit-sizes of the vector that is currently under scope.

2.3 A quasi-linear-time reduction algorithm

As a broad approximation, L^3, L^2 and H-LLL are generalisations of Euclid's greatest common divisor algorithm. The successive bases computed during the execution play the role of Euclid's remainders, and the elementary matrix operations performed on the bases play the role of Euclid's quotients. L^3 may be interpreted in such a framework. It is slow because it computes its "quotients" using all the bits from the "remainders" rather than the most significant bits only: The cost of computing one Euclidean division in an L^3 way is $O(\beta^{1+\varepsilon})$, leading to an overall $O(\beta^{2+\varepsilon})$ bound for Euclid's algorithm (for β-long input integers). Lehmer [64] proposed an acceleration of Euclid's algorithm by the means of truncations. Since the ℓ most significant bits of the remainders provide the first $\Omega(\ell)$ bits of the sequence of quotients, one may: Truncate the remainders to precision ℓ; Compute the sequence of quotients for the truncated remainders; Store the first $\Omega(\ell)$ bits of the quotients into an $\Omega(\ell)$-bit matrix; Apply the latter to the input remainders, which are shortened by $\Omega(\ell)$ bits; And iterate. The cost gain stems from the decrease of the bit-lengths of the computed remainders. Choosing $\ell \approx \sqrt{\beta}$ leads to a complexity bound of $O(\beta^{3/2+\varepsilon})$. In the early 1970's, Knuth [62] and Schönhage [108] independently observed that using Lehmer's idea recursively leads to a gcd algorithm with complexity bound $O(\beta^{1+\varepsilon})$. The above approach for the computation of gcds has been successfully adapted to two-dimensional lattices [122, 110, 25], and the resulting algorithm was then used in [27] to reduce lattices in arbitrary dimensions in quasi-linear time. Unfortunately, the best known cost bound for the latter is $O(\beta^{1+\varepsilon}(\log\beta)^{n-1})$ for fixed n.

$\widetilde{L}^1$ aims at adapting the Lehmer-Knuth-Schönhage gcd framework to the case of LLL-reduction. $\widetilde{L}^1$ takes as inputs LLL parameters Ξ and a non-singular $B \in \mathbb{Z}^{n\times n}$; terminates within $O(n^{5+\varepsilon}\beta + n^{4+\varepsilon}\beta^{1+\varepsilon})$ bit operations, where $\beta = \log\max\|\mathbf{b}_i\|$; and returns a basis of the lattice spanned by B which is Ξ-LLL-reduced.

The efficiency of the fast gcd algorithms stems from two sources: Performing operations on truncated remainders is meaningful (which allows one to consider remainders with smaller bit-sizes), and the obtained transformations corresponding to the quotients sequence have small bit-sizes (which allows one to transmit at low cost the information obtained on the truncated remainders back to the genuine remainders). We achieve an analogue of the latter by *gradually feeding the input* to the reduction algorithm, and the former is ensured thanks to the modified notion of LLL-reduction which is *resilient to truncations*. The main difficulty in adapting the fast gcd framework lies in the multi-dimensionality of lattice reduction. In particular, the basis vectors may have significantly differing magnitudes. This means that basis truncations must be performed column-wise. Also, the resulting unimodular transformations may have large magnitudes, hence need to be truncated for being be stored on few bits.

To handle these difficulties, we focused on reducing bases which are a mere scalar shift from being reduced. We call this process *lift-reducing*, and it can be used to provide a family of new reduction algorithms. Lift-reducing was introduced by Belabas [13], van Hoeij and Novocin [52], in the context of specific lattice bases that are encountered while factoring rational polynomials (e.g., with the algorithm from [51]): It was restricted to reducing specific sub-lattices which avoid the above dimensionality difficulty. We generalise these results to the following. Suppose that we wish to reduce a matrix B with the property that $B_0 := \sigma_\ell^{-k} B$ is reduced for some k and σ_ℓ is the diagonal matrix $\mathrm{diag}(2^\ell, 1, \ldots, 1)$. If one runs L^3 on B directly then the structure of B_0 is not being exploited. Instead, the matrix B can be slowly

reduced allowing us to control and understand the intermediate transformations: Compute the unimodular transform U_1 (with any reduction algorithm) such that $\sigma_\ell B_0 U_1$ is reduced and repeat until we have $\sigma_\ell^k B_0 U_1 \cdots U_k = B(U_1 \cdots U_k)$. Each entry of U_i and each entry of $U_1 \cdots U_i$ can be bounded sensitive to the shape of the lattice (i.e., to k).

The algorithm from Figure 2.6 shows how to LLL-reduce an arbitrary lattice basis given a Lift-reducing algorithm (used in Step 5).

Inputs: LLL parameters Ξ; a non-singular $B \in \mathbb{Z}^{n\times n}$.
Output: A Ξ-reduced basis of $L(B)$.
1. $B := \mathrm{HNF}(B)$.
2. For k from $n-1$ down to 1 do
3. Let C be the bottom-right $(n-k+1)$-dimensional submatrix of B.
4. $\ell_k := \lceil \log_2(b_{kk}) \rceil$, $C := \sigma_{\ell_k}^{-1} C$.
5. Find U' unimodular such that $\sigma_{\ell_k} C U'$ is Ξ-reduced.
6. Let U be the block-diagonal matrix $\mathrm{diag}(I, U')$.
7. Compute $B := B \cdot U$, reducing row i symmetrically modulo b_{ii} for $i < k$.
8. Return B.

Figure 2.6: Reducing LLL-reduction to lift-reduction.

Lemma 1 *The algorithm of Figure 2.6 Ξ-reduces B such that* $\max \|\mathbf{b}_i\| \leq 2^\beta$ *using*

$$O\left(n^{4+\varepsilon}(\beta^{1+\varepsilon} + n)\right) + \sum_{k=n-1}^{1} \mathcal{C}_k$$

bit operations, where $\mathcal{C}_k$ is the cost of Step 5 for the specific value of k.

The above shows that we can now restrict ourselves to Lift-reducing efficiently. In order to be able to Lift-reduce by means of truncations, we can use the sensitivity analysis of Section 2.1 along with a bound on the coefficients of a lift-reducing U.

Lemma 2 *Let Ξ_1, Ξ_2 be valid parameters. Let $\ell \geq 0$, $B \in \mathbb{R}^{n\times n}$ (with R-factor R) be Ξ_1-reduced and U such that $C = \sigma_\ell BU$ (with R-factor R') is Ξ_2-reduced. We have:*

$$\forall i, j: \ |u_{ij}| \leq \zeta^n \cdot \frac{r'_{jj}}{r_{ii}} \leq 2^\ell \zeta^{2n} \cdot \frac{r_{jj}}{r_{ii}},$$

for some ζ that depends only on Ξ_1 and Ξ_2.

Suppose the sequence of the r_{ii}'s is very unbalanced. As B is reduced, this can only occur when the sequence increases sharply. In that situation, Lemma 2 does not prevent U from being arbitrarily large. However, its entries may be truncated while preserving unimodularity and the fact that it actually lift-reduces B.

Lemma 3 *Let Ξ_1, Ξ_2, Ξ_3 be valid LLL parameters with $\Xi_2 > \Xi_3$. There exists a constant c such that the following holds for any $\ell \geq 0$. Let $B \in \mathbb{R}^{n\times n}$ (with R-factor R) be Ξ_1-reduced, and U be unimodular such that $\sigma_\ell BU$ (with R-factor R') is Ξ_2-reduced. If $\Delta U \in \mathbb{Z}^{n\times n}$ satisfies $|\Delta u_{ij}| \leq 2^{-(\ell + c\cdot n)} \cdot \frac{r'_{j,j}}{r_{i,i}}$ for all i, j, then $U + \Delta U$ is unimodular and $\sigma_\ell B(U + \Delta U)$ is Ξ_3-reduced.*

Lifting and truncation are the main conceptual ingredients for the `Lift`-$\widetilde{L}^1$ algorithm, given in Figure 2.7. `Lift`-$\widetilde{L}^1$ makes use of specific compact representations of basis and transformation matrices to handle the possible unbalancedness of the current basis vectors. `Lift`-$\widetilde{L}^1$ makes use of several subroutines: The `BaseCase` algorithm performs lift-reduction for small values of ℓ and relies on a truncation and a call to H-LLL (see Section 2.2); `BaseCase` may be used with $\ell = 0$ to strengthen the reducedness of a reduced basis (i.e., Ξ_2-reducing a Ξ_1-reduced basis, for $\Xi_2 > \Xi_1$); The MSB_k function replaces a matrix B by a truncated $B + \Delta B$ with $\max \frac{\|\Delta \mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-k}$; the $U_1 \odot U_2$ operation is a matrix multiplication of U_1 and U_2 which is specifically designed to handle the specific format chosen for the unimodular transformations (in particular, it performs a truncation after computing the product, to ensure that the output entries have small bit-sizes).

Inputs: Valid LLL-parameters $\Xi_3 > \Xi_2 \geq \Xi_4 > \Xi_1$; a lifting target ℓ; $(B', (e_i)_i)$ such that $B = B' \cdot \mathrm{diag}(2^{e_i}) \in \mathbb{Q}^{n \times n}$ is Ξ_1-reduced and $\max |b'_{ij}| \leq 2^{\ell + c \cdot n}$ for some $c > 0$.
Output: $(U', (d_i)_i, x)$ such that $\sigma_\ell BU$ is Ξ_1-reduced, with $U = 2^{-x}\mathrm{diag}(2^{-d_i}) \cdot U' \cdot \mathrm{diag}(2^{d_i})$ and $\max |u'_{ij}| \leq 2^{2\ell + 2c \cdot n}$.

1. If $\ell \leq n$, then use `BaseCase` with lifting target ℓ. Otherwise:
2. */* Prepare 1st recursive call */*
   Call `BaseCase` on (B, Ξ_2); Let U_1 be the output.
3. $B_1 := \mathrm{MSB}_{(\ell/2 + c_3 \cdot n)}(B \cdot U_1)$.
4. */* 1st recursive call */*
   Call `Lift`-$\widetilde{L}^1$ on B_1, with lifting target $\ell/2$; Let U_{R_1} be the output.
5. */* Prepare 2nd recursive call */*
   $U_{1R_1} := U_1 \odot U_{R_1}$.
6. $B_2 := \sigma_{\ell/2} BU_{1R_1}$.
7. Call `BaseCase` on (B_2, Ξ_3). Let U_2 be the output.
8. $U_{1R_12} := U_{1R_1} \odot U_2$.
9. $B_3 := \mathrm{MSB}_{(\ell/2 + c_3 \cdot n)}(\sigma_{\ell/2} BU_{1R_12})$.
10. */* 2nd recursive call */*
    Call `Lift`-$\widetilde{L}^1$ on B_3, with lifting target $\ell/2$; Let U_{R_2} be the output.
11. */* Prepare output */*
    $U_{1R_12R_2} := U_{1R_12} \odot U_{R_2}$.
12. $B_4 := \sigma_\ell BU_{1R_12R_2}$.
13. Call `BaseCase` on (B_4, Ξ_4); Let U_3 be the output.
14. $U := U_{1R_12R_2} \odot U_3$; Return U.

Figure 2.7: The `Lift`-$\widetilde{L}^1$ algorithm.

The $\widetilde{\mathrm{L}}^1$ algorithm is the algorithm from Figure 2.6, where `Lift`-$\widetilde{\mathrm{L}}^1$ is used to implement lift-reduction (with appropriate pre- and post-processings to handle the input and output formats of `Lift`-$\widetilde{\mathrm{L}}^1$. A careful bit-operation count involving an amortisation analysis (over the successive calls to `Lift`-$\widetilde{\mathrm{L}}^1$ leads to the following result.

Theorem 6 ([88]) *Given as inputs Ξ and a matrix $B \in \mathbb{Z}^{n \times n}$ with $\max \|\mathbf{b}_i\| \leq 2^\beta$, the $\widetilde{L}^1$ algorithm returns a Ξ-reduced basis of $L(B)$ within $O(n^{5+\varepsilon}\beta + n^{4+\varepsilon}\beta^{1+\varepsilon})$ bit operations.*

2.4 Perspectives

The complexity of the $\widetilde{\mathrm{L}}^1$ algorithm with respect to $\beta = \log \max \|\mathbf{b}_i\|$ seems hard to improve further: Up to a constant factor, it is the same as for the best known gcd algorithms [62, 108], i.e., $O(\mathcal{M}(\beta)\log\beta)$, where $\mathcal{M}(\ell)$ denotes the time required to multiply two ℓ-bit long integers. The remaining challenge on the cost of LLL-reduction consists in decreasing the dependences in the lattice dimension n.

Let ω denote the fast linear algebra exponent: Two n-dimensional square matrices over a field K may be multiplied within $O(n^\omega)$ arithmetic operations over K (the Coppersmith and Winograd algorithm [22] achieves $\omega \leq 2.376$). Then the complexity of $\widetilde{\mathrm{L}}^1$ is $O(n^{5+\varepsilon}\beta + n^{\omega+1+\varepsilon}\beta^{1+\varepsilon})$. Intuitively, the first term corresponds to $O(\beta)$ LLL-reductions of n-dimensional matrices whose entries have bit-sizes $O(n)$ and that perform $O(n^2)$ LLL swaps, whereas the second term corresponds to the binary tree multiplication of $O(\beta)$ matrices of dimension n and whose entries have bit-sizes $O(n)$ (this originates from Steps 1 and 7 of the algorithm of Figure 2.6). It seems the second term is intrinsic to $\widetilde{\mathrm{L}}^1$, and that a new reduction approach is required for avoiding it. The first term, which currently dominates the overall cost, could however be improved using techniques developed by Schönhage, Koy and Schnorr and Storjohann [109, 63, 121] to lower the number of arithmetic operations arising from the size-reductions. It remains to be seen whether these techniques can be combined with the numerical analysis and floating-point arithmetic approaches used in L^2 and H-LLL. Furthermore, even if the latter difficulty can be handled, and if no further progress is made on the numerical analysis aspects, the required floating-point precision will remain $\Omega(n)$: If R is the R-factor of an LLL-reduced matrix, the quantity $\mathrm{cond}(R)$ from Theorem 2 can be as large as $2^{\Omega(n)}$ (see [20, Re. 7]), which can be compensated only by taking a working precision that is $\Omega(n)$.

From the discussion above, it appears that more work is required on the numerical aspects of LLL. A first step consists in assessing whether what has been achieved for L^2 and H-LLL can be carried over to [109, 63, 121]. This will hopefully allows the complexity of $\widetilde{\mathrm{L}}^1$ to be decreased down to $\widetilde{O}(n^{\omega+1}\beta)$. To decrease this bit-complexity further, significantly new ingredients will be needed, in particular to avoid the $\Omega(n)$-bit-long floating-point arithmetic, at least for most arithmetic operations.

Independently from the cost objective, the techniques developed for $\widetilde{\mathrm{L}}^1$ could prove useful for related computational tasks. Can they be exploited for reduction of polynomial matrices [90, 79] or for Hermite Normal For computations? Also, the lifting technique of $\widetilde{\mathrm{L}}^1$ seems reminiscent of the PSLQ algorithm for disclosing integer relations between real numbers [29]: By revisiting PSLQ under this new light, one might be able to prove its correctness under floating-point arithmetic and to investigate its bit-complexity.

CHAPTER 3

Stronger Lattice Reduction Algorithms

The LLL lattice reduction algorithm and its variants run in polynomial time but only provide vectors that are no more than exponentially longer (with respect to the lattice dimension n) than the shortest non-zero lattice vectors. This worst-case behaviour seems to also hold in practice [83], up to a constant factor in the exponent.

Solving the Shortest and Closest Vectors Problem exactly is much more expensive. There exist three main families of SVP and CVP solvers, which we compare in Table 3.1. (In the table, and more generally in the present chapter introduction, we omit the arithmetic costs, which are all $\mathrm{poly}(n, \max \log \|\mathbf{b}_i\|)$, where $(\mathbf{b}_i)_i \mathbb{Z}^{n \times n}$ is the input basis.) The algorithm by Micciancio and Voulgaris [76, 75] aims at computing the Voronoi cell of the lattice, whose knowledge facilitates the tasks of solving SVP and CVP. This algorithm allows one to solve SVP and CVP deterministically, in time $\leq 2^{2n+o(n)}$ and space $\leq 2^{n+o(n)}$.

Single exponential time complexity had already been achieved about 10 years before by Ajtai, Kumar and Sivakumar [8, 9], with an algorithm that consists in saturating the space with a cloud of (perturbed) lattice points. But the saturation algorithms suffer from at least three drawbacks: They are Monte Carlo (their success probability can be made exponentially close to 1, though); The CVP variants of these algorithms may only find vectors that are no more than $1 + \varepsilon$ times further away from the target than the optimal solution(s) (it is possible to choose an arbitrary $\varepsilon > 0$, but the complexity grows quickly when ε tends to 0); and their best known complexity upper bounds are higher than that of the Micciancio-Voulgaris algorithm relying on the Voronoi cell computation. The Ajtai *et al.* SVP solver has been

Table 3.1: Comparing the three main families of SVP and CVP solvers.

	Time complexity upper bound	Space complexity upper bound	Underlying principle
[76, 75] for SVP and CVP	$2^{2n+o(n)}$	$2^{n+o(n)}$	Voronoi cell
[8, 97, 87, 77, 96] for SVP [9, 14] for $\mathrm{CVP}_{1+\varepsilon}$	$2^{2.465n+o(n)}$ $(2+1/\varepsilon)^{O(n)}$	$2^{1.325n+o(n)}$ $(2+1/\varepsilon)^{O(n)}$	Saturation
[30, 31, 59, 60, 48, 44] for SVP [30, 31, 59, 60, 48, 44] for CVP	$n^{n/(2e)+o(n)}$ $n^{n/2+o(n)}$	$\mathrm{poly}(n)$ $\mathrm{poly}(n)$	Enumeration

successively improved in [97, 87, 77, 96], and the currently best time complexity upper bound is $2^{2.465n+o(n)}$, with a space requirement bounded by $2^{1.325n+o(n)}$. Improvements on the Ajtai *et al.* CVP solver have been proposed by Blömer and Naewe [14].

Before the elaboration of the saturation-based solvers by Ajtai, Kumar and Sivakumar, the asymptotically fastest SVP and CVP solvers relied on a deterministic procedure that enumerates all lattice vectors within a prescribed distance to a given target vector (chosen to be $\mathbf{0}$ in the case of SVP). This procedure exploits the Gram-Schmidt orthogonalisation of the input basis to recursively bound the integer coordinates of the candidate solutions. Enumeration-based SVP and CVP solvers were first described by Fincke and Pohst [30, 31] and Kannan [59, 60]. Kannan used it to propose solvers with bit-complexities $n^{O(n)}$. These were later refined by Helfrich [48].

The practicality of SVP solvers has attracted much attention, as it is the dominating cost component of the generic cryptanalyses of the lattice-based cryptographic schemes. Determining and extrapolating the current practical limits is crucial for choosing key sizes that are meaningful for desired security levels. For currently handleable dimensions, the enumeration-based SVP solvers seem to outperform those of the other families. This statement requires clarification, as rigorous codes providing correctness guarantees can be accelerated significantly by allowing heuristics, which makes the comparison task more complex. On the rigorous side, all the available implementations providing strong correctness guarantees (e.g., `fplll` [16] or the SVP solvers of the Magma computational algebra system [15]) rely on the enumeration process. They seem to be currently limited to dimensions around 75. On the heuristic side, the solvers of the saturation and enumeration families can be accelerated by making reasonable but unproved assumptions. The heuristic implementations of the enumeration families, relying on tree pruning strategies [106, 107, 120, 36], seem to outperform the heuristic implementations of the saturation families [87, 77]. They seem to allow one to reach dimensions around 110. The enumeration solvers have also been implemented in hardware [49, 23]. At the time being, the Micciancio-Voulgaris algorithm relying on the Voronoi cell seems uncompetitive, and would require further practical investigation.

With Guillaume Hanrot, we studied in detail the cost of the enumeration procedure of the enumeration-based solvers, in order to get a better grasp on the currently most practical family of SVP and CVP solvers. This line of work will be described in Section 3.1. We decreases the best known complexity upper bounds of Kannan's SVP solver (resp. CVP solver) from $n^{n/2+o(n)}$ (resp. $n^{n+o(n)}$) to $n^{n/(2e)+o(n)}$ (resp. $n^{n/2+o(n)}$). The ideas underlying this result are summarised in Section 3.1.

When the dimension of the lattice under scope is too high, all known SVP and CVP solvers (and thus also HKZ reduction) become prohibitively expensive. However, it is still possible to compute lattice bases of higher quality than those provided by LLL-type algorithms. Schnorr's hierarchy [103] of reduction algorithms allows one to achieve a continuum between the LLL and HKZ reductions. The best known theoretical variant, in terms of achieved basis quality for any fixed computational cost, is due to Gama and Nguyen [34]. All known realizations of Schnorr's hierarchy (see the surveys [80, 102]) rely on an algorithm that solves SVP for smaller-dimensional lattices. We let β denote the largest dimension in which the SVP solver is used. Table 3.2 describes the time/quality trade-off reached by Schnorr's hierarchy. In this table, the output quality is measured by the best known Hermite factor upper bound of an output basis, where the Hermite factor of a basis $(\mathbf{b}_i)_{i\leq n}$ of a lattice L is defined as $HF((\mathbf{b}_i)_{i\leq n}) = \|\mathbf{b}_1\|/(\det L)^{1/n}$.

Table 3.2: Time/quality trade-offs reached by several reduction algorithms.

	HKZ	[34] with parameter β	LLL
Hermite factor	$\sqrt{n}$	$\sqrt{\beta(1+\varepsilon)}^{\frac{n-1}{\beta-1}}$	$2^{O(n)}$
Time	$2^{O(n)}$	$2^{O(\beta)} \cdot \mathrm{poly}(n)$	$\mathrm{poly}(n)$

In practice, the heuristic and somewhat mysterious BKZ algorithm from [106] is used instead of the slide reduction algorithm from [34] (see [35] for a detailed account on the practical behaviour of BKZ).

With Guillaume Hanrot and Xavier Pujol, we started trying to analyse the BKZ algorithm, in order to understand why it performs so well in practice. Our results so far remain partial. However, we could provide the first non-trivial worst-case analysis on the performance of BKZ: We showed that if stopped after a polynomial number of calls to the underlying low-dimensional SVP solver, the Hermite factor of the output basis admits a bound similar to that of the basis returned by the algorithm from [34]. We elaborate on this result in Section 3.2.

3.1 Cost analysis of the enumeration-based SVP and CVP solvers

The `Enum` algorithm, given in Figure 3.1, enumerates $L \cap \mathcal{B}_n(\mathbf{t}, A)$ by using the triangular relationship between the basis $(\mathbf{b}_i)_{i \leq n}$ of L and its Gram-Schmidt orthogonalisation $(\mathbf{b}_i^*)_{i \leq n}$. More precisely, it relies on the two following observations:

- If $\mathbf{x} = \sum_i x_i \mathbf{b}_i$ belongs to $L \cap \mathcal{B}_n(\mathbf{t}, A)$, then, for any $i \leq n$, we have $\mathbf{x}^{(i)} \in L^{(i)} \cap \mathcal{B}_{n-i+1}(\mathbf{t}^{(i)}, A)$, where $\mathbf{x}^{(i)}$, $L^{(i)}$ and $\mathbf{t}^{(i)}$ are the projections of $\mathbf{x}$, L and $\mathbf{t}$ respectively, orthogonally to the linear span of $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$.
- Enumerating $L^{(n)} \cap \mathcal{B}_1(\mathbf{t}^{(n)}, A)$ is easy and once $L^{(i+1)} \cap \mathcal{B}_{n-i}(\mathbf{t}^{(i+1)}, A)$ is known, it is easy to enumerate $L^{(i)} \cap \mathcal{B}_{n-i+1}(\mathbf{t}^{(i)}, A)$: Assume that $\mathbf{x}^{(i)} \in L^{(i)} \cap \mathcal{B}_{n-i+1}(\mathbf{t}^{(i)}, A)$; Write $\mathbf{x}^{(i)} = \mathbf{x}^{(i+1)} + (x_i + c_i)\mathbf{b}_i^*$ for some $x_i \in \mathbb{Z}$ and $c_i \in \mathbb{Q}$; Once $\mathbf{x}^{(i+1)} \in L^{(i+1)} \cap \mathcal{B}_{n-i}(\mathbf{t}^{(i+1)}, A)$ is fixed, we must have

$$x_i \in \mathbb{Z} \cap \left[-c_i - \frac{\sqrt{A^2 - \|\mathbf{x}^{(i+1)}\|^2}}{\|\mathbf{b}_i^*\|}, -c_i + \frac{\sqrt{A^2 - \|\mathbf{x}^{(i+1)}\|^2}}{\|\mathbf{b}_i^*\|} \right] \tag{3.1}$$

These observations lead to interpreting `Enum` as a depth-first tree traversal, where the nodes correspond to the considered $(x_n, \ldots, x_i)$ for all i, and the sons of a node $(x_n, \ldots, x_{i+1})$ are the $(x'_n, \ldots, x'_i)$ such that $x_j = x'_j$ for all $j \geq i+1$. The execution starts at the nodes $()$ (i.e., the node whose sons are the (x_n)'s for the possible values of x_n), and the goal is to obtain the list of the tree leaves $(x_n, \ldots, x_1)$.

Algorithm `Enum` may be used directly to solve SVP and CVP, once the bound A has been set. In the case of SVP, it may be derived from Minkowski's theorem, or from the current basis $(\mathbf{b}_i)_{i \leq n}$: For example, one may choose $A = \min(\min_i \|\mathbf{b}_i\|, \sqrt{\gamma_n}(\det L)^{1/n})$.

Inputs: A basis $(\mathbf{b}_i)_{i\le n}$ of a lattice $L \subseteq \mathbb{Q}^{n\times n}$, $\mathbf{t} \in \mathbb{Q}^n$, $A > 0$.
Output: All vectors in $L \cap \mathcal{B}(\mathbf{t}, A)$.
1. Compute the $\mu_{i,j}$'s and $\|\mathbf{b}_i^*\|^2$'s.
2. Compute the t_i's such that $\mathbf{t} = \sum_i t_i \mathbf{b}_i^*$.
3. $S := \{\}$, $\ell := 0$, $\mathbf{x} := \mathbf{0}$, $x_n := \lceil t_n - A/\|\mathbf{b}_n^*\| \rceil$, $i := n$.
4. While $i \le n$, do
5. $\quad \ell_i := (x_i - t_i + \sum_{j>i} x_j \mu_{ji})^2 \|\mathbf{b}_i^*\|^2$,
6. $\quad$ If $i = 1$ and $\sum_{1\le j\le n} \ell_j \le A^2$, $S := S \cup \{\mathbf{x}\}$, $x_1 := x_1 + 1$.
7. $\quad$ If $i \ne 1$ and $\sum_{j\ge i} \ell_j \le A^2$, $i := i - 1$, $x_i := \left\lceil t_i - \sum_{j>i}(x_j \mu_{ji}) - \sqrt{\frac{A^2 - \sum_{j>i} \ell_j}{\|\mathbf{b}_i^*\|^2}} \right\rceil$.
8. $\quad$ If $\sum_{j\ge i} \ell_j > A$, then $i := i + 1$, $x_i := x_i + 1$.
9. Return S.

Figure 3.1: The `Enum` algorithm.

In the case of CVP, it may be derived from any bound on the covering radius $\rho(L)$, such as $\frac{1}{2}\sqrt{\sum_i \|\mathbf{b}_i^*\|^2}$. The bound may also be set heuristically using the Gaussian heuristic: The guess for A is then derived from the equation $\mathrm{vol}(\mathcal{B}_n(\mathbf{t}, A)) \approx \det(L)$, and is increased if no solution is found. The bound A can also be decreased during the execution of `Enum`, every time a better solution is found. Also, the space required by `Enum` may be more than $\mathrm{poly}(n, \log \max \|\mathbf{b}_i\|)$, because $|S|$ might be exponentially large. The space requirement can be made $\mathrm{poly}(n, \log \max \|\mathbf{b}_i\|)$ for the SVP and CVP applications, as only a single shortest/closest vector is required: The update of S in `Enum` should then be replaced by an update of the best solution found so far.

During its execution, algorithm `Enum` considers all points in $L^{(i)} \cap \mathcal{B}_{n-i+1}(\mathbf{t}^{(i)}, A)$, for $i = n, n-1, \ldots, 1$. An inherent drawback is that the complexity may be (significantly) more than $|L \cap \mathcal{B}_n(\mathbf{t}, A)|$. This is because it often occurs that at some stage, an element of $L^{(i+1)} \cap \mathcal{B}_{n-i}(\mathbf{t}^{(i+1)}, A)$ has no descendant in $L^{(i)} \cap \mathcal{B}_{n-i+1}(\mathbf{t}^{(i)}, A)$ (i.e., the interval in Equation (3.1) contains no integer): This corresponds to a "dead-end" in the enumeration tree.

The cost of `Enum` can be bounded by $\sum_i |L^{(i)} \cap \mathcal{B}_{n-i+1}(\mathbf{t}^{(i)}, A)|$, up to a small polynomial factor. The Gaussian heuristic allows us to estimate the latter quantity: If K is a measurable subset of the span of the n-dimensional lattice L, then $|K \cap L| \approx \mathrm{vol}(K)/\det(L)$ (where vol denotes the n-dimensional volume). This leads to the approximation (for $i \le n$):

$$|L^{(i)} \cap \mathcal{B}_{n-i+1}(\mathbf{t}^{(i)}, A)| \approx \frac{2^{O(n)} A^{n-i+1}}{(n-i+1)^{\frac{n-i+1}{2}} \cdot \prod_{j=i}^{n} \|\mathbf{b}_j^*\|}.$$

This heuristic cost analysis of the enumeration process, given in [44], has interesting practical implications:

- It allows a user to assess in advance if the computation has a chance to terminate within a reasonable amount of time. This has been implemented in the Magma computational algebra system [15].
- Suppose the tree search corresponding to `Enum` is performed using parallel processors. The heuristic cost formula above can be used to estimate the sizes of subtrees, in order to give well-balanced tasks to slave processors [23].

- Finally, this formula can be tweaked to account for tree pruning and thus to optimise the pruning strategy [36, 120].

Unfortunately, from a theoretical standpoint, some of the involved balls are very small compared to their corresponding lattice $L^{(i)}$, and it seems hard to prove that the heuristic is indeed valid in these cases. Though of mostly theoretical nature (because of the fuzzy $2^{O(n)}$ factor), the following result provides theoretical evidence towards the validity of the Gaussian heuristic in the present situation.

Theorem 7 ([44]) *If given as inputs a lattice basis $(\mathbf{b}_i)_{i\leq n}$ and a target vector $\mathbf{t}$, the number of arithmetic operations performed during the execution of* `Enum` *can be bounded from above by:*

$$2^{O(n)} \prod_{1\leq i\leq n} \max\left(1, \frac{A}{\sqrt{n}\|\mathbf{b}_i^*\|}\right) \leq 2^{O(n)} \max_{I\subseteq[1,n]} \left(\frac{A^{|I|}}{\sqrt{n}^{|I|}\cdot\prod_{i\in I}\|\mathbf{b}_i^*\|}\right).$$

The latter upper bound for the cost of `Enum` and the heuristic cost estimate strongly depend on A and on the decrease of the $\|\mathbf{b}_i^*\|$'s. This suggests that the more reduced the basis $(\mathbf{b}_i)_i$, the lower the cost. Fincke and Pohst [30] initially used a LLL-reduced basis $(\mathbf{b}_i)_i$. For such a basis, we have $\|\mathbf{b}_{i+1}^*\| \geq \|\mathbf{b}_i^*\|/2$ for all i, which leads to a $2^{O(n^2)}$ complexity upper bound. Kannan [59] observed that the cost of `Enum` is so high that a much more aggressive pre-processing significantly lowers the total cost while negligibly contributing to it. Kannan's SVP algorithm is in fact an HKZ-reduction algorithm that calls itself recursively in lower dimensions to strengthen the reducedness before calling `Enum`. The bases $(\mathbf{b}_i)_i$ given as inputs to `Enum` always satisfy the following conditions: It is size-reduced, $\|\mathbf{b}_2^*\| \geq \|\mathbf{b}_1^*\|/2$ and once projected orthogonally to $\mathbf{b}_1$, the other $\mathbf{b}_i$'s are HKZ-reduced. We call such bases *quasi-HKZ-reduced*. A detailed analysis gives that if a basis $(\mathbf{b}_i)_{i\leq n}$ is quasi-HKZ-reduced, then:

$$\max_{I\subseteq[1,n]} \left(\frac{\|\mathbf{b}_1\|^{|I|}}{\sqrt{n}^{|I|}\cdot\prod_{i\in I}\|\mathbf{b}_i^*\|}\right) \leq 2^{O(n)} n^{n/(2e)}.$$

The calls to `Enum` dominate the overall cost of Kannan's HKZ-reduction algorithm, so that Kannan's SVP solver terminates within $n^{n/(2e)+o(n)}$ arithmetic operations. Kannan's CVP algorithm first HKZ-reduces the given lattice basis, and then calls `Enum` using the reduced basis. The number of arithmetic operations it performs can be bounded from above by $n^{n/2+o(n)}$.

The cost upper bound of Kannan's SVP algorithm is optimal. More precisely, a probabilistic construction due to Ajtai [6, 7] can be adapted to prove the existence of HKZ-reduced bases for which `Enum` actually performs $n^{n/(2e)+o(n)}$ bit operations [45]. The proof relies on the following converse to Theorem 7.

Theorem 8 ([46, Se. 3]) *If given as inputs a lattice basis $(\mathbf{b}_i)_{i\leq n}$ and a target vector $\mathbf{t}$, the number of arithmetic operations performed during the execution of* `Enum` *can be bounded from below by:*

$$2^{O(n)} \prod_{i=i_0}^{n} \frac{A}{\sqrt{n}\|\mathbf{b}_i^*\|},$$

where i_0 is the smallest such that $\max_{i\geq i_0}\|\mathbf{b}_i^*\| \leq \frac{2}{3}\sqrt{\frac{A}{n}}$.

For CVP, a gap remains between the lowest known complexity upper bound $n^{n/2+o(n)}$ for Kannan's solver and its largest known worst-case complexity lower bound $n^{n/(2e)+o(n)}$.

3.2 Terminating the Schnorr-Euchner BKZ algorithm

As mentioned at the beginning of this chapter, slide reduction [34] seems to be outperformed by the BKZ algorithm [35] in practice: For comparable run-times, the quality of the computed bases seems higher with BKZ (or, equivalently, the same basis quality is reached faster with BKZ). With respect to run-time, no reasonable bound was known on the number of calls to the β-dimensional HKZ reduction algorithm it needs to make before termination (a naive bound $O(\beta)^n$ can be proven if BKZ is slightly modified, see [43, App. A]). In practice, this number of calls does not seem to be polynomially bounded [35] 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 [107, 81].

Figure 3.2 illustrates the evolution of the Hermite factor during the execution of the original BKZ and modified BKZ' (described in Figure 3.3). We refer the reader to [43] for a description of the (mild) differences between BKZ and BKZ'. The corresponding experiment is as follows: We generated 64 "knapsack-like" lattice bases [83] of dimension $n = 108$, with non-trivial entries of bit-lengths $100n$; Each was LLL-reduced using `fplll` [16] (with parameters $\delta = 0.99$ and $\eta = 0.51$); Then for each we ran NTL's BKZ [114] and an implementation of BKZ' in NTL, with blocksize 24. Figure 3.2 only shows the beginning of the executions (more than half were more than 6 times longer). A "tour" corresponds to calling the smaller dimensional HKZ-reduction algorithm $n - \beta + 1$ times. As can be observed, BKZ and BKZ' quickly end up spending of lot of time making very little progress.

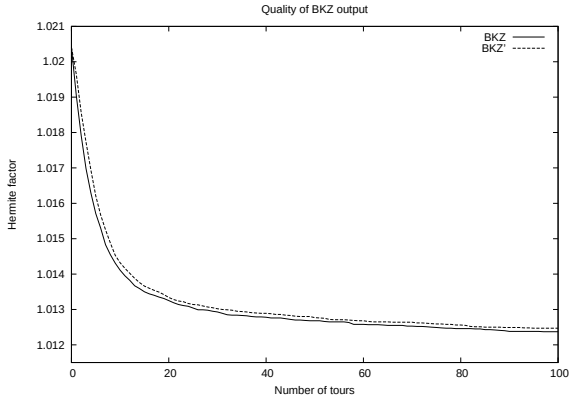

Figure 3.2: Evolution of the Hermite factor $\frac{\|\mathbf{b}_1\|}{(\det L)^{1/n}}$ during the execution of BKZ and BKZ'.

With Xavier Pujol and Guillaume Hanrot, we showed that if terminated within polynomially many calls to HKZ/SVP, a slightly modified version of BKZ returns bases of excellent quality, close to that reached by the slide reduction algorithm.

Theorem 9 *There exists $C > 0$ such that the following holds for all n and β. Let $B = (\mathbf{b}_i)_{i \leq n}$ be a basis of a lattice L, given as input to the modified BKZ algorithm of Figure 3.3*

with block-size β. If terminated after $C\frac{n^3}{\beta^2}\left(\log n + \log\log \max_i \frac{\|\mathbf{b}_i\|}{(\det L)^{1/n}}\right)$ calls to an HKZ-reduction (or SVP solver) in dimension β, the output $(\mathbf{c}_i)_{i\leq n}$ is a basis of L that satisfies (with $\gamma'_\beta \leq \beta$ defined as the maximum of Hermite's constants in dimensions $\leq \beta$):

$$\|\mathbf{c}_1\| \leq 2(\gamma'_\beta)^{\frac{n-1}{2(\beta-1)}+\frac{3}{2}} \cdot (\det L)^{\frac{1}{n}}.$$

If L is a rational lattice, then the overall cost is $\leq \mathrm{poly}(n, \log\max\|\mathbf{b}_i\|)\cdot \mathcal{C}_{\mathtt{HKZ}}(\beta)$, where $\mathcal{C}_{\mathtt{HKZ}}(\beta) = 2^{O(\beta)}$ is any upper bound on the time complexity of HKZ-reducing a β-dimensional lattice basis of bit-size $\leq \mathrm{poly}(\beta)$.

Input: A basis $(\mathbf{b}_i)_{i\leq n}$ and a blocksize β.
Output: A basis of $L[(\mathbf{b}_i)_{i\leq n}]$.
1. Repeat while no change occurs or termination is requested:
2. For $k \leftarrow 1$ to $n-\beta+1$,
3. Modify $(\mathbf{b}_i)_{k\leq i\leq k+\beta-1}$ so that $(\mathbf{b}_i^{(k)})_{k\leq i\leq k+\beta-1}$ is HKZ-reduced,
4. Size-reduce $(\mathbf{b}_i)_{i\leq n}$.

Figure 3.3: The modified BKZ algorithm: BKZ'.

To achieve this result, we used a new approach for analysing lattice reduction algorithms. The classical approach to bound their runtimes was to introduce a quantity, sometimes called potential, involving the current Gram-Schmidt norms $\|\mathbf{b}_i^*\|$, which always strictly decreases every time some elementary step is performed. This technique was introduced by Lenstra, Lenstra and Lovász [65] for analysing their LLL algorithm, and is still used in all complexity analyses of (current variants of) LLL. It was later adapted to stronger lattice reduction algorithms [103, 33, 102, 34]. We still measure progress with the $\|\mathbf{b}_i^*\|$'s, but instead of considering a single scalar combining them all, we look at the *full vector* $(\|\mathbf{b}_i^*\|)_{i\leq n}$. More specifically, we observe that each call to HKZ within BKZ has the effect of applying an affine transformation to the vector $(\log\|\mathbf{b}_i^*\|)_{i\leq n}$: Instead of providing a lower bound to the progress made on a "potential", we are then led to analyse 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 $(\log\|\mathbf{b}_i^*\|)_{i\leq n}$ is to essentially replace β consecutive coefficients by their average. We formalise this intuition by making the following *Heuristic Sandpile Model Assumption (SMA)*: We assume for any HKZ-reduced basis $(\mathbf{b}_i)_{i\leq\beta}$, we have $x_i = \frac{1}{2}\log\gamma_{\beta-i+1} + \frac{1}{\beta-i+1}\sum_{j=i}^{\beta} x_j$ for all $i \leq \beta$, with $\mathbf{x} = (\log\|\mathbf{b}_i^*\|)_{i\leq\beta}$. Under this assumption, the execution of BKZ exactly matches with a dynamical system that can be explicited and fully analysed. A BKZ tour corresponds to applying a specific affine transformation to $\mathbf{x}$: $\mathbf{x} \leftarrow A\mathbf{x}+\mathbf{\Gamma}$. The fixed-points of A provide information on the output quality of BKZ, whereas the largest singular value of A^TA smaller than 1 drives the speed of convergence.

However, the heuristic SMA is not always correct: Consider for example orthogonal $\mathbf{b}_i$'s of growing norms. This difficulty can be circumvented by considering the vector $(\mu_i)_{i\leq n}$ where $\mu_i = \frac{1}{i}\sum_{j=1}^{i}\log\|\mathbf{b}_j^*\|$ for any i. This amortisation was already used in [44] for analysing HKZ-reduced bases. Here it allowed us to *rigorously* bound the evolution of $(\mu_i)_{i\leq n}$ by the

orbit of a vector under another dynamical system. This bound holds coefficient-wise, and relies on the result below.

Lemma 4 ([44, Le. 3]) *If $(\mathbf{b}_i)_{i\leq\beta}$ is HKZ-reduced, then*

$$\forall k \leq \beta,\ \mu_k - \mu_\beta \leq \frac{\beta-k}{k}\log\Gamma_\beta(k),$$

with $\Gamma_\beta(k) = \sum_{i=\beta-k}^{\beta-1}\frac{\log\gamma_{i+1}}{2i}$.

This new dynamical system bounding the evolution of $(\mu_i)_{i\leq n}$ happens to be a slight modification of the dynamical system used in the idealised sandpile model, and the analysis performed for the idealised model can be adapted to the rigorous set-up.

3.3 Conclusion and perspectives

Many important techniques and results on solving SVP and CVP have been discovered in the last few years: The Ajtai *et al.* saturation-based solver [8] was obtained 10 years ago and has steadily been improved since then, while the Micciancio-Voulgaris Voronoi-based [76] solver is even more recent. The interest in this topic was revived at least in large part thanks to the rise of lattice-based cryptography: Assessing the precise limits of the algorithms for SVP, CVP and their approximations is the key towards providing meaningful key-sizes ensuring specific security levels.

The saturation-based and Voronoi-based algorithms have better asymptotic complexity bounds than the enumeration-based solvers, but in practice this comparison is reversed. It is tempting to investigate this oddity. Is it possible to improve these algorithms further? Are there reasonable heuristics that would allow for competing with heuristic enumeration-based solvers? For example, saturation-based solvers make use of perturbations to hide information to the inner sieving steps. It is unclear whether the perturbations of the lattice vectors in saturation-based solvers are inherently necessary or just an artifact of the proof. As these perturbations lead to increased complexity bounds, proving them unnecessary could make these solvers competitive with [76]. Also, is it a valid heuristic to remove them in practice? It is also completely conceivable that faster solvers exist, that remain to be discovered. For example, is it possible to achieve exponential time complexity with a polynomially bounded space requirement? Are there ways to exploit quantum computations to obtain better complexity bounds? An important challenge in this line of research would be to design a polynomial-time algorithm that could find non-zero lattice vectors that are no more than polynomially longer (in the dimension) than the lattice minimum. In particular, this could render lattice-based cryptography insecure.

The newer types of efficient SVP and CVP solvers seem to at least partially circumvent lattice reduction: The Ajtai *et al.* solver only uses a LLL-type algorithm and the Voronoi-based Micciancio-Voulgaris uses strong reduction only to improve the constant in the exponent of its complexity bound, whereas the cost of the enumeration is highly dependent on the strongness of the reduction of the input basis. This raises the question of the relevance of lattice reduction in the first place. An important step towards assessing this relevance consists in determining whether a BKZ-like trade-off between cost and smallness of the computed

vectors could be achieved (or even beaten) without lattice reduction. For example, is it possible to accelerate the Ajtai *et al.* and Micciancio-Voulgaris algorithms, without lowering the output quality too much?

Finally, even if the tasks of improving LLL-type algorithms and SVP/CVP solvers seem quite distinct, the works described in Chapters 2 and 3 suggest a few possible links. Naturally, it is tempting to exploit the analysis of the BKZ algorithm based on dynamical systems to simplify and maybe improve the block-based algorithms for fast LLL-type reduction [109, 63, 121]. In the other direction, the lift-reduction strategy developed for the $\widetilde{L}^1$ of Section 2.3 could be investigated in the context of solving SVP. At a very high level, it consists in finding a sequence of small deformation steps such that: The start of the deformation path is already handled (in the case of $\widetilde{L}^1$, a reduced basis of some lattice); The ending point of the deformation path contains the solution of the problem under scope (in the case of $\widetilde{L}^1$, a reduced basis of the input lattice); And each deformation step is computationally easy. In the case of SVP, this suggests starting from an easy lattice and progressively deforming it towards the desired lattice, so that each step is cheaper to solve than a general instance of SVP.

CHAPTER 4

Asymptotically Efficient Lattice-Based Encryption Schemes

The aim of an encryption scheme is to securely transmit information between two parties. An asymmetric, or public-key, encryption scheme allows anyone to encrypt a message using the receiver's public key, while only the receiver can decrypt messages encrypted under its public key, using the associated secret key. As opposed to symmetric encryption, asymmetric encryption does not require the parties to have previously agreed on a shared secret key. Asymmetric encryption schemes were first proposed at the end of the 1970's [101, 70]. Most public-key encryption schemes deployed today heuristically/provably rely on the assumption that (a variation of) one of the following problems is hard to solve:

- The integer factorisation problem: Given an integer N which is the product of two large primes, factor N.
- The discrete logarithm problem in finite fields (DLP). Given a finite field $\mathbb{F}$, a generator g of the group of units $\mathbb{F}^\times$ and an element $h \in \mathbb{F}^\times$, find $x \in \mathbb{Z}$ such that $h = g^x$.
- The discrete logarithm problem in elliptic curves (ECDLP). Given an elliptic curve E over a finite field, a generator g of a large subgroup of E and an element h in that subgroup, find $x \in \mathbb{Z}$ such that $h = x \cdot g$.

It is worth noting that the actual hardness assumptions that are made involve average instances for specific input distributions: Typically, DLP and ECDLP involve a random h, while IF involves random prime factors.

All known encryption schemes relying on these problems suffer from at least two main drawbacks. First, they are inherently slow. The operations that are performed for encryption and decryption, such as modular exponentiation, typically cost $O(n^3)$ in naive arithmetic or $O(n^{2+\varepsilon})$ using fast integer multiplication, where n is the bit-size of the key pair. Further, in the case of IF and DLP (and also for ECDLP for the curves used in pairing-based cryptography), the best known attacks are sub-exponential with respect to the key-length: They can typically be mounted with $2^{\widetilde{O}(n^{1/3})}$ bit operations. In order to resist to attacks costing up to 2^t (we call t the *security parameter*), then n should be set $\widetilde{\Omega}(t^3)$, making encryption and decryption typically cost $\widetilde{\Omega}(t^6)$. Second, the fact that these problems can all be solved in polynomial-time using a quantum computer [112, 113] raises the question whether they might not share some common weakness, even against classical computers. Further, many schemes are proved secure under the assumptions that ad-hoc variants of IF, DLP

and ECDLP are hard, creating a myriad of related but not so clearly equivalent hardness assumptions.

A few other mathematical objects and corresponding algorithmic problems seem to enable cryptographic constructions without some of the drawbacks mentioned above. These include error correcting codes and systems of multivariate polynomial equations. However, the natural problems on Euclidean lattices seem to be the most promising candidates. On the one hand, schemes based on lattices have very low asymptotic complexities (they typically involve basic linear algebra operations, over small rings), which can be lowered even further using specific subfamilies of lattices (see below). On the other hand, these schemes admit security proofs under a small number of well-identified worst-case problems (as opposed to average-case hardness assumptions for specific input distributions). Additionally, lattice-based cryptographic primitives involve simple and flexible operations: this flexibility allows for the design of primitives that were not realized before, such as fully homomorphic encryption [38].

Lattice-based encryption comes in two flavours: practical with heuristic security arguments, and slower but with very strong security proofs. From a practical perspective, the `NTRUEncrypt` scheme offers impressive encryption and decryption performances. It was devised by Hoffstein, Pipher and Silverman, and first presented at the Crypto'96 rump session [54]. Although its description relies on arithmetic over the polynomial ring $\mathbb{Z}_q[x]/(x^n - 1)$ for n prime and q a small power of 2 (we use the notation $\mathbb{Z}_q$ to denote the ring of integers modulo q), it was quickly observed that breaking it could be expressed as a problem over Euclidean lattices [21]. At the ANTS'98 conference, the NTRU authors gave an improved presentation including a thorough assessment of its practical security against lattice attacks [55]. We refer to [53] for an up-to-date account on the past 15 years of security and performance analyses. Nowadays, `NTRUEncrypt` is generally considered as a reasonable alternative to the encryption schemes based on IF, DLP and ECDLP, as testified by its inclusion in the IEEE P1363 standard [56]. It is also often considered as the most viable post-quantum public-key encryption (see, e.g., [94]).

In parallel to a rising number of attacks and practical improvements on `NTRUEncrypt` the (mainly) theoretical field of provably secure lattice-based cryptography has steadily been developed. It originated in 1996 with Ajtai's acclaimed worst-case to average-case reduction [3], leading to a collision-resistant hash function that is as hard to break as solving several worst-case problems defined over lattices. Ajtai's average-case problem is now referred to as the *Small Integer Solution* problem (SIS). Another major breakthrough in this field was the introduction in 2005 of the *Learning with Errors* problem (LWE) by Regev [98, 99]: LWE is both hard on the average (worst-case lattice problems quantumly reduce to it), and sufficiently flexible to allow for the design of cryptographic functions. In the last few years, many cryptographic schemes have been introduced that are provably at least as secure as LWE and SIS are hard (and thus provably secure, assuming the worst-case hardness of lattice problems). These include encryption schemes secure under Chosen Plaintext Attacks and Chosen Ciphertext Attacks, identity-based encryption schemes, digital signatures, *etc* (see [99, 91, 39, 17, 1] among others, and the surveys [74, 100]).

The currently easiest (and most efficient) way to build encryption schemes whose security relies on the worst-case hardness of standard lattice problems (such as SIVP_γ for approximation factors γ that are polynomial in n) is to proceed via the LWE problem. To formulate it, we need the following notation: For an $\mathbf{s} \in \mathbb{Z}_q^n$, and a distribution χ over $\mathbb{Z}_q$, we let $D_{\mathbf{s},\chi}$

denote the distribution over $\mathbb{Z}_q^{n+1}$ obtained by sampling $\mathbf{a} \hookleftarrow U(\mathbb{Z}_q^n)$ and $e \hookleftarrow \chi$ and returning $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$. The *Computational Learning With Errors Problem* Comp-LWE$_{q,\chi}$ is as follows: Given n and an access to an oracle that samples from $D_{\mathbf{s},\chi}$ for some $\mathbf{s} \in \mathbb{Z}_q^n$, find $\mathbf{s}$. The *Decisional Learning With Errors Problem* Dec-LWE$_{q,\chi}$ is as follows: Let $\mathbf{s} \hookleftarrow U(\mathbb{Z}_q^n)$; Given access to an oracle $\mathcal{O}$ which is sampling from either $U(\mathbb{Z}_q^{n+1})$ or $D_{\mathbf{s},\chi}$, decide in which situation we are. Regev showed that if χ is the Gaussian distribution of standard deviation αq reduced modulo q and rounded to the closest integer (which we denote by χ_α), then:

- If $\gamma, q \geq \omega(\sqrt{n}/\alpha)$ (resp. $\gamma, q \geq \Omega(n/\alpha)$, then there exists a quantum polynomial-time (resp. sub-exponential-time) reduction from SIVP$_\gamma$ to Comp-LWE$_{q,\chi_\alpha}$.

- If $q \leq \text{poly}(n)$ (resp. $q \leq 2^{o(n)}$) is prime, then there exists a randomised polynomial-time (resp. sub-exponential-time) reduction from Comp-LWE$_{q,\chi_\alpha}$ to Dec-LWE$_{q,\chi_\alpha}$.

When the number m of calls to the oracle is predetermined, then LWE has a natural linear algebra interpretation. Comp-LWE consists in finding $\mathbf{s} \in \mathbb{Z}_q^m$ from $(A, A\mathbf{s} + \mathbf{e})$, where $A \hookleftarrow U(\mathbb{Z}_q^{m \times n})$ and $\mathbf{e} \hookleftarrow \chi^m$, while stating that Dec-LWE is hard to solve means that for $\mathbf{s} \hookleftarrow U(\mathbb{Z}_q^n)$, the distributions $U(\mathbb{Z}_q^{m \times (n+1)})$ and $(A, A\mathbf{s} + \mathbf{e})$, with $A \hookleftarrow U(\mathbb{Z}_q^{m \times n})$, are computationally indistinguishable.

Ajtai [5] showed how to simultaneously sample, in polynomial-time, an LWE matrix $A \in \mathbb{Z}_q^{m \times n}$ and a (trapdoor) basis $S = (\mathbf{s}_1, \ldots, \mathbf{s}_m) \in \mathbb{Z}^{m \times m}$ of the lattice $A^\perp = \{\mathbf{b} \in \mathbb{Z}^m : \mathbf{b}^T A = \mathbf{0} \bmod q\}$, with the following properties: The distribution of A is within exponentially small statistical distance to $U(\mathbb{Z}_q^{m \times n})$; The basis vectors $\mathbf{s}_1, \ldots, \mathbf{s}_m$ are short. Recently, Alwen and Peikert [10, 11] improved Ajtai's construction in the sense that the created basis has shorter vectors: They achieved $\|S\| = O(r\sqrt{m})$ with $m = \Omega(n \frac{\log^2 q}{\log r})$ for any integer r.

These results allow for the elegant design of a cryptosystem that is provably secure under Chosen Plaintext Attacks [39, 91]:

- Key Generation: Run the Alwen-Peikert algorithm and obtain a pair $(A, S) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times m}$; Sample $A' \hookleftarrow U(\mathbb{Z}_q^{m \times n})$ and let (A, A') be the public key while S is the secret key;

- Encryption: To encrypt $\mathbf{M} \in \{0,1\}^m$, sample $\mathbf{s} \hookleftarrow U(\mathbb{Z}_q^n)$ and $\mathbf{e}, \mathbf{e}' \hookleftarrow \chi^m$, and return $(A\mathbf{s} + \mathbf{e}, A'\mathbf{s} + \mathbf{e}' + \lfloor q/2 \rfloor \mathbf{M})$;

- Decryption: To decrypt $(\mathbf{C}_1, \mathbf{C}_2) \in \mathbb{Z}_q^m \times \mathbb{Z}_q^m$, first compute $S\mathbf{C}_1 \bmod q$, which should be exactly $S\mathbf{e}$ (over the integers), since the entries of both S and $\mathbf{e}$ are small with respect to q; Then recover $\mathbf{e}$ by multiplying by S^{-1} and then recover $\mathbf{s}$; Using $\mathbf{C}_2$ and $\mathbf{s}$, recover $\mathbf{e}' + \lfloor q/2 \rfloor \mathbf{M}$; At this stage, the vector $\mathbf{M}$ can be recovered componentwise by assessing whether the given component is close to $q/2$ or to 0.

Unfortunately, this encryption scheme is bound to remain somewhat inefficient, as the key-size is $\Omega(m^2 \log q) = \Omega(n^2)$. In this chapter, we present two ways of waiving this restriction and obtaining quasi-optimal efficiency: The key-size and the run-times of encryption and decryption all will be $\widetilde{O}(t)$, where t is the security parameter (i.e., all known attacks cost $2^{\Omega(t)}$).

4.1 A first attempt, from a trapdoor one-way function

In order to accelerate encryption schemes based on lattices, Micciancio [71] introduced the class of structured *cyclic* lattices, which correspond to ideals in polynomial rings $\mathbb{Z}[x]/(x^n - 1)$, and presented the first provably secure one-way function based on the worst-case hardness of the restriction of poly(n)-SVP to cyclic lattices. At the same time, thanks to its algebraic structure, this one-way function enjoys high efficiency: $\widetilde{O}(n)$ evaluation time and storage cost. Subsequently, Lyubashevsky and Micciancio [68] and independently Peikert and Rosen [92] showed how to modify Micciancio's function to construct an efficient and provably secure collision resistant hash function. For this, they introduced the more general class of *ideal* lattices, which correspond to ideals in polynomial rings $\mathbb{Z}[x]/f(x)$ (via the isomorphism that consists in identifying a polynomial to its coefficient vector). In this chapter, we will restrict ourselves to $f(x) = x^n + 1$ with n a power of 2 (this is the $2n$-th cyclotomic polynomial, and $\mathbb{Z}[x]/(x^n + 1)$ is the ring of integers of the $2n$-th cyclotomic number field). The collision resistance relies on the hardness of the restriction of poly(n)-SVP to ideal lattices (called poly(n)-Ideal-SVP). The average-case collision-finding problem is a natural computational problem called Ring-SIS, which has been shown to be as hard as the worst-case instances of Ideal-SVP.

The Small Integer Solution problem with parameters q, m, β (SIS$_{q,m,\beta}$) is as follows: Given n and a matrix A sampled uniformly in $\mathbb{Z}_q^{m\times n}$, find $\mathbf{e} \in \mathbb{Z}^m \setminus \{\mathbf{0}\}$ such that $\mathbf{e}^T A = \mathbf{0} \bmod q$ (the modulus being taken component-wise) and $\|\mathbf{e}\| \leq \beta$. The Ring Small Integer Solution problem with parameters q, m, β and f (Id-SIS$^f_{q,m,\beta}$) is as follows: Given n and m polynomials $g_1, \ldots, g_m$ chosen uniformly and independently in $\mathbb{Z}_q[x]/f$, find $e_1, \ldots, e_m \in \mathbb{Z}[x]$ not all zero such that $\sum_{i\leq m} e_i g_i = 0$ in $\mathbb{Z}_q[x]/f$ and $\|\mathbf{e}\| \leq \beta$, where $\mathbf{e}$ is the vector obtained by concatenating the coefficients of the e_i's. Id-SIS is exactly SIS, where G is chosen to be $\mathrm{rot}_f(\mathbf{g})$. The matrix $\mathrm{rot}_f(\mathbf{g})$ is defined as follows: If $r \in \mathbb{Z}[x]/f$, then $\mathrm{rot}_f(r) \in \mathbb{Q}^{n\times n}$ is the matrix whose rows are the $x^i r(x) \bmod f(x)$'s, for $0 \leq i < n$; This is extended to the matrices A over $\mathbb{Q}[x]/f$, by applying rot_f component-wise.

Our construction attempts to use a variant of LWE using a structured matrix A instead of $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$. More specifically, The Ideal Learning With Errors problem Comp-Id-LWE$_{q,m,\chi}$ is the same as Comp-LWE restricted to m calls to the oracle $D_{\mathbf{s},\chi}$, except that $A = \mathrm{rot}_f(\mathbf{a})$ with $\mathbf{a} \hookleftarrow U((\mathbb{Z}_q[x]/f)^m)$. The space saving due to using Id-LWE arises from the fact that n rows of A may be stored with n elements of $\mathbb{Z}_q$ instead of n^2. This allows us to set Id-LWE's m to be n times smaller than LWE's m. The efficiency improvement arises from the fact that a multiplication $\mathrm{rot}_f(g) \cdot \mathbf{b}$ may be performed in quasi-linear time, as the coefficients of the obtained vector are those of the polynomial $b(x) \cdot g(1/x) \mod x^n + 1$, which may be computed efficiently using fast polynomial multiplication [37, Ch. 8]. However, it is not straightforward to adapt Regev's reductions from worst-case lattice problems to Dec-LWE, to this structured setting (although this has been recently achieved by Lyubashevsky, Peikert and Regev [69], as explained in the next section). To circumvent this difficulty, we proposed a new reduction, directly from Id-SIS to Id-LWE, using Regev's quantum reduction:

Theorem 10 *Let q, m, n be integers with $q \equiv 3 \bmod 8$, $n \geq 32$ a power of 2, $\mathrm{poly}(n) \geq m \geq 41 \log q$ and $\alpha < \min\left(\frac{1}{10\sqrt{\ln(10m)}}, 0.006\right)$. Let χ_α be the normal law of standard deviation αq, reduced modulo q and rounded to the closest integer. Suppose that there exists an algorithm that solves Comp-Id-LWE$_{q,m,\chi}$ in time T and with probability $\varepsilon \geq 4m \exp\left(-\frac{\pi}{4\alpha^2}\right)$.*

Then there exists a quantum algorithm that solves Id-SIS$_{q,m,\frac{\sqrt{m}}{2\alpha}}$ *in time* $\mathrm{poly}(T,n)$ *and with probability* $\frac{\varepsilon^3}{64} - O(\varepsilon^5) - 2^{-\Omega(n)}$.

This result ensures that Comp-Id-LWE is indeed at least as hard to solve as worst-case lattice problems for ideal lattices, because Id-SIS is known to be so [68, 92]. However, it is weaker than what could hope from a full-fledged adaptation of Regev's worst-case to average-case reduction, for two reasons: First, Comp-Id-LWE is restricted to a fixed m, and second it is not clear how to derive from the result above that a decisional variant of Comp-Id-LWE is also hard.

However, an asymptotically efficient encryption scheme can still be built. At this stage, the hardness of Comp-Id-LWE provides us a family of one-way functions: $\mathbf{s} \mapsto \mathrm{rot}_f(\mathbf{a}) \cdot \mathbf{s} + \mathbf{e}$. Furthermore, the Ajtai-Alwen-Peikert trapdoor construction for LWE can be adapted to derive a a family of trapdoor one-way functions, see [119]. By combining this trapdoor function with the Goldreich-Levin generic hardcore function [40, Sec. 2.5] we obtain a security proof for the following encryption scheme Id-Enc.

- **Key generation.** For security parameter n, run the modified Ajtai-Alwen-Peikert algorithm from [119] to get $\mathbf{g} \in (\mathbb{Z}_q[x]/(x^n+1))^m$ and a trapdoor S (such that $S \cdot g = \mathbf{0}$ in $\mathbb{Z}_q[x]/(x^n+1)$). Let $\ell_I = O(n \log q) = \widetilde{O}(n)$, generate $\mathbf{r} \in \mathbb{Z}_2^{\ell_I + \ell_M}$ uniformly and define the Toeplitz matrix $M_{GL} \in \mathbb{Z}_2^{\ell_M \times \ell_i}$ (allowing fast multiplication [89]) whose i-th row is $[r_i, \ldots, r_{\ell_I + i - 1}]$. The public key is $(\mathbf{g}, \mathbf{r})$ and the secret key is S.

- **Encryption.** Given ℓ_M-bit message M with $\ell_M = n/\log n = \widetilde{\Omega}(n)$ and public key $(\mathbf{g}, \mathbf{r})$, sample $(\mathbf{s}, \mathbf{e})$ with $\mathbf{s} \in \mathbb{Z}_q^n$ uniform and $\mathbf{e}$ sampled from χ_α, and evaluate $C_1 = \mathrm{rot}_f(\mathbf{g})^T \cdot \mathbf{s} + \mathbf{e}$. Compute $C_2 = M \oplus (M_{GL} \cdot \mathbf{s})$, where $\mathbf{s}$ is viewed as a string over $\mathbb{Z}_2^{\ell_I}$, the product $M_{GL} \cdot \mathbf{s}$ is computed over $\mathbb{Z}_2$, and the $\oplus$ notation stands for the bit-wise XOR function. Return the ciphertext (C_1, C_2).

- **Decryption.** Given ciphertext (C_1, C_2) and secret key $(S, \mathbf{r})$, invert C_1 to compute $(\mathbf{s}, \mathbf{e})$ such that $\mathrm{rot}_f(\mathbf{g})^T \cdot \mathbf{s} + \mathbf{e} = C_1$, and return $M = C_2 \oplus (M_{GL} \cdot \mathbf{s})$.

Theorem 11 *Any chosen plaintext attack against indistinguishability of Id-Enc with runtime T and success probability $1/2 + \varepsilon$ provides an algorithm for Id-LWE$^f_{q,m,\chi_\alpha}$ with runtime $O(2^{3\ell_M} n^3 \varepsilon^{-3} \cdot T)$ and success probability $\Omega(2^{-\ell_M} n^{-1} \cdot \varepsilon)$.*

4.2 A security proof for NTRUEncrypt

Last year, Lyubashevsky, Peikert and Regev [69] proposed in a concurrent and independent work a full-fledged adaptation of Regev's reductions for Dec-LWE, to the case of structured lattices. To define the Decisional Ring Learning With Errors Problem (Dec-RLWE), we first need a few notations.

Let $R = \mathbb{Z}[x]/(x^n+1)$ for n a power of 2 and $R_q = \mathbb{Z}_q[x]/(x^n+1) = R/(qR)$, for an integer q. For $s \in R_q$ and ψ a distribution in R_q, we define $A_{s,\psi}$ as the distribution obtained by sampling the pair $(a, as+e)$ with $(a,e) \hookleftarrow U(R_q) \times \psi$. The (parametrised) distributions ψ_α used by Lyubashevsky et al are a bit technical to define, but may be thought of as n-dimensional Gaussian vectors with standard deviations αq, rounded to the closest

integer vector and reduced modulo q. They actually differ a little from this: For instance, the distribution ψ_α is itself chosen randomly, from a (parametrised) distribution Y_α. The important facts to be remembered are that sampling from Y_α and from the sample ψ_α can be performed in quasi-linear time (with respect to $n \log q$), and that the samples from ψ_α are small (smaller than $\alpha q \sqrt{n} \omega(\sqrt{\log n})$ with overwhelming probability) and can be obtained in quasi-linear time (with respect to $n \log q$).

The *Ring Learning With Errors Problem* with parameters q and α (Dec-RLWE$_{q,\alpha}$) is as follows. Let $\psi \hookleftarrow \overline{\mathrm{Y}}_\alpha$ and $s \hookleftarrow U(R_q)$. Given access to an oracle $\mathcal{O}$ that produces samples in $R_q \times R_q$, distinguish whether $\mathcal{O}$ outputs samples from $A_{s,\psi}$ or from $U(R_q \times R_q)$. The distinguishing advantage should be $1/\mathrm{poly}(n)$ (resp. $2^{-o(n)}$) over the randomness of the input, the randomness of the samples and the internal randomness of the algorithm. It was shown in [69] that there exists a randomised polynomial-time (resp. sub-exponential) quantum reduction from γ-Id-SVP to Dec-RLWE$_{q,\alpha}$, with $\gamma = \omega(n^{1.5} \log n)/\alpha$ (resp. $\Omega(n^{2.5})/\alpha$), under the assumptions that: $\alpha q = \omega(n\sqrt{\log n})$ (resp. $\Omega(n^{1.5})$) with $\alpha \in (0,1)$; and $q = \mathrm{poly}(n)$ is prime such that $x^n + 1$ has n distinct linear factors modulo q.

With Ron Steinfeld, we exploited the proven hardness of the Dec-RLWE problem to modify `NTRUEncrypt` so that it becomes provably secure, under the assumed quantum hardness of standard worst-case lattice problems, restricted to ideal lattices. The revised scheme `NTRUEncrypt'` is as follows.

- **Key generation.** Sample f' from $D_{\mathbb{Z}^n,\sigma}$ using the Gentry et al. sampler (Theorem 1); Let $f = 2f' + 1$ and restart if f is not invertible in R_q. Similarly, sample g from $U(R_q^\times)$. The secret key is f, while the public key is $h = 2g/f \in R_q^\times$.
- **Encryption.** Given message $M \in R$ whose coefficients belong to $\{0,1\}$, set $s, e \hookleftarrow \phi_\alpha \hookleftarrow \mathrm{Y}_\alpha$ and return ciphertext $C = hs + 2e + M \in R_q$.
- **Decryption.** Given ciphertext C and secret key f, compute $C' = f \cdot C \in R_q$ and return C' mod 2.

The scheme is very similar to `NTRUEncrypt`, apart from minor-looking differences which have significant impact for allowing for a security proof based on the hardness of Dec-RLWE.

1. In `NTRUEncrypt`, the polynomial rings are $R^{NTRU} = \mathbb{Z}[x]/(x^n - 1)$ with n a prime number, and $R_q^{NTRU} = \mathbb{Z}_q[x]/(x^n - 1)$ with q a power of 2. These rings were modified to match those for which Dec-RLWE is known to be hard.
2. As a side effect, the modification of q allows for setting NTRU's p to 2 (in the original scheme, p was chosen to be $x + 2$ or 3, because it is required to be invertible modulo q.
3. In `NTRUEncrypt`, the secret key polynomial f' and g were chosen with coefficients in $\{-1, 0, 1\}$, with predetermined numbers of coefficients being set to 0. Instead, we sample f' and g using discrete Gaussians over R, rejecting the samples that are not invertible in R_q. This allows us for showing that f/g is statistically close to uniform over $R_q^\times$.
4. In `NTRUEncrypt`, no error term e is used in the encryption algorithm, and the nonce s is chosen from a distribution similar to that of f'. Adding the error allows for relying on the hardness of Dec-RLWE.

By relying on fast arithmetic over polynomials, we obtain that the encryption and decryption operations of NTRUEncrypt' can be performed in time quasi-linear in n. Furthermore, the key generation process is also very efficient, as the rejection probability is small: The probability that $x \hookleftarrow U(R_q)$ is not invertible modulo n is $O(n/q)$, and this fact can also be shown to hold when $x \hookleftarrow D_{\mathbb{Z}^n,\sigma}$ for a sufficiently large σ.

The security of NTRUEncrypt' relies on a mild modification of Dec-RLWE. First, using [12, Le. 2], it is possible to show that Dec-RLWE remains hard if s is sampled from ψ_α (instead of $s \hookleftarrow U(R_q)$). Furthermore, the problem still remains hard if we assume that the a of $(a, as+e)$ is sampled from $U(R_q^\times)$ instead of $U(R_q)$, because there are sufficiently many invertible elements in R_q. Given these modifications on Dec-RLWE, and the fact that $2 \in R_q^\times$, it follows that if h was sampled uniformly in $R_q^\times$, then a ciphertext $2(hs+e)+M$ would be indistinguishable from uniform. This is our main contribution: We show that if h is sampled as described, its statistical distance to uniformity is exponentially small. Overall, this leads to the following result.

Theorem 12 *Suppose n is a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q = \mathrm{poly}(n)$ such that $q^{\frac{1}{2}-\varepsilon} = \omega(n^{2.5}\log^2 n)$ (resp. $q^{\frac{1}{2}-\varepsilon} = \omega(n^3 \log^{1.5} n)$), for arbitrary $\varepsilon \in (0, 1/2)$. Let $\sigma = 2n\sqrt{\ln(8nq)} \cdot q^{\frac{1}{2}+\varepsilon}$ and $\alpha^{-1} = \omega(n^{0.5}\log n\sigma)$. If there exists an Chosen Plaintext Attack against the Indistinguishability of* NTRUEncrypt' *which runs in time $T = \mathrm{poly}(n)$ and has success probability $1/2 + 1/\mathrm{poly}(n)$ (resp. time $T = 2^{o(n)}$ and success probability $1/2 + 2^{-o(n)}$), then there exists a $\mathrm{poly}(n)$-time (resp. $2^{o(n)}$-time) quantum algorithm for γ-Id-SVP with $\gamma = O(n^3 \log^{2.5} n q^{\frac{1}{2}+\varepsilon})$ (resp. $\gamma = O(n^4 \log^{1.5} n q^{\frac{1}{2}+\varepsilon})$). Moreover, the decryption algorithm succeeds with probability $1 - n^{-\omega(1)}$ over the choice of the encryption randomness.*

As mentioned above, the most important fact that remains to be proven is that the public key polynomial is indeed close to uniformly distributed in $R_q^\times$. We denote by $D_{\sigma,z}^\times$ the discrete Gaussian $D_{\mathbb{Z}^n,\sigma}$ restricted to $R_q^\times + z$, where z is an arbitrary element of R_q. The public key uniformity is a direct consequence of the following result.

Theorem 13 *Let $n \geq 8$ be a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q \geq 5$. Let $\varepsilon > 0$ and $\sigma \geq 2n\sqrt{\ln(8nq)} \cdot q^{\frac{1}{2}+2\varepsilon}$. Let $p \in R_q^\times$, $y_i \in R_q$ and $z_i = -y_i p^{-1} \bmod q$ for $i \in \{1,2\}$. Then*

$$\Delta\left[\frac{y_1 + p \cdot D_{\sigma,z_1}^\times}{y_2 + p \cdot D_{\sigma,z_2}^\times} \bmod q \; ; \; U\left(R_q^\times\right)\right] \leq 2^{3n} q^{-\lfloor \varepsilon n \rfloor}.$$

The proof consists in showing that for every $a \in R_q^\times$, the probability that $f_1/f_2 = a$ is extremely close to $(q-1)^{-n}$, where $f_i \hookleftarrow y_1 + p \cdot D_{\sigma,z_1}^\times$. For this, it suffices to show that for every $a_1, a_2 \in R_q^\times$, the probability that $f_1 a_1 + f_2 a_2 = 0$ is extremely close to $(q-1)^{-n}$. The fact that f_1 and f_2 are not sampled with rejection is handled via an inclusion-exclusion argument. From now on, we assume for simplicity that $f_1, f_2 \hookleftarrow D_{\mathbb{Z}^n,\sigma}$. It then suffices to bound the statistical distance to $U(R_q^\times \times R_q^\times \times R_q)$ of the triple $(a_1, a_2, f_1 a_1 + f_2 a_2)$ when $a_i \hookleftarrow U(R_q^\times)$ and $f_i \hookleftarrow D_{\mathbb{Z}^n,\sigma}$. The latter question is reminiscent of the left-over hash lemma [57], and a bound can be obtained in this specific context using standard tools on discrete Gaussians [73, 39] (and some elementary algebraic number theory). The reader is referred to [118] for more details.

4.3 Perspectives

Replacing arbitrary lattices by ideal lattices and unstructured matrices by structured matrices was a significant step towards making lattice-based cryptography practical. However, its deployment remains curbed by a few important difficulties. First and perhaps most importantly, the practical limits of the best known attacks are still fuzzy. At the time this document is being written, the statement from [35] that solving γ-SVP with $\gamma = (1.01)^n$ is hard with current implementations seems generally accepted. However, it gives no precise estimate of how hard it actually is, nor how it would extrapolate for different levels of security.

From a security viewpoint, the restriction to ideal lattices further narrows the link to NP-hardness results. The LWE and SIS problems were already only known to be no easier than γ-SIVP and γ-CVP for values of γ for which no NP-hardness result is known to hold. In fact, it is even strongly suspected that these problem relaxations are not NP-hard, as they belong to NP$\cap$coNP [2]. But in the case of ideal lattices, no NP-hardness result is known to hold even for $\gamma = 1$. On the other hand, there is no known significant computational advantage when standard lattice problems are restricted to ideal lattices (apart from the gap decisional version of SVP). The assumption that the restriction to ideal lattices creates no vulnerability needs further investigation. On a related topic, the argument that lattice-based cryptography (including schemes based on ideal lattices) resists would-be quantum computers needs further backing. For the moment, it relies on the single observation that it is not known how to exploit quantum computing to solve standard lattice problems significantly more efficiently than with classical computers. Proving a quantum hardness result (such as QMA-hardness, the quantum equivalent to NP-hardness) for a lattice problem would substantiate the assumption.

Finally, cryptography is far from being restricted to encryption resisting to Chosen Plaintext Attacks. Far more functionalities and efficient implementations thereof would be required if lattice-based cryptography were to be deployed widely. There has already been quite some effort spent on signatures (see, e.g., [67]) and hash functions [68, 93]. On the other hand, at the time being there is no lattice-based encryption scheme both resisting Chosen Ciphertext Attacks and consisting of quasi-linear time algorithms. An interesting goal in this context would be to discover an equivalent to pairings on elliptic curves in the context of lattices, as these have allowed for the efficient realization of many cryptographic functionalities.

Bibliography

[1] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *Proceedings of Eurocrypt*, volume 6110 of *LNCS*, pages 553–572. Springer, 2010.

[2] D. Aharonov and O. Regev. Lattice problems in NP ∩ coNP. *J. ACM*, 52(5):749–765, 2005.

[3] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of STOC*, pages 99–108. ACM, 1996.

[4] M. Ajtai. The shortest vector problem in l_2 is NP-hard for randomized reductions (extended abstract). In *Proceedings of STOC*, pages 284–293. ACM, 1998.

[5] M. Ajtai. Generating hard instances of the short basis problem. In *Proceedings of ICALP*, volume 1644 of *LNCS*, pages 1–9. Springer, 1999.

[6] M. Ajtai. The worst-case behavior of Schnorr's algorithm approximating the shortest nonzero vector in a lattice. In *Proceedings of STOC*, pages 396–406. ACM, 2003.

[7] M. Ajtai. Optimal lower bounds for the Korkine-Zolotareff parameters of a lattice and for Schnorr's algorithm for the shortest vector problem. *Theory of Computing*, 4(1):21–51, 2008.

[8] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings of STOC*, pages 601–610. ACM, 2001.

[9] M. Ajtai, R. Kumar, and D. Sivakumar. Sampling short lattice vectors and the closest lattice vector problem. In *Proceedings of CCC*, pages 53–57, 2002.

[10] J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. In *Proceedings of STACS*, LNCS, pages 75–86. Springer, 2009.

[11] J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. *Theory of Computing Systems*, 48(3):535–553, 2011.

[12] B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Proceedings of CRYPTO*, volume 5677 of *LNCS*, pages 595–618. Springer, 2009.

[13] K. Belabas. A relative van Hoeij algorithm over number fields. *Journal of Symbolic Computation*, 37(5):641–668, 2004.

[14] J. Blömer and S. Naewe. Sampling methods for shortest vectors, closest vectors and successive minima. *Theor. Comput. Science*, 410(18):1648–1665, 2009.

[15] W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system. I. The user language. *Journal of Symbolic Computation*, 24(3–4):235–265, 1997. `http://magma.maths.usyd.edu.au/magma/`.

[16] D. Cadé, X. Pujol, and D. Stehlé. fplll-3.1, a floating-point LLL implementation. `http://perso.ens-lyon.fr/xavier.pujol/fplll`.

[17] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *Proceedings of Eurocrypt*, volume 6110 of *LNCS*, pages 523–552. Springer, 2010.

[18] X.-W. Chang and C. C. Paige. Componentwise perturbation analyses for the QR factorization. *Numerische Mathematik*, 88:319–345, 2001.

[19] X.-W. Chang, C. C. Paige, and G. W. Stewart. Perturbation analyses for the QR factorization. *SIAM J. Matrix Anal. Appl.*, 18:775–791, 1997.

[20] X.-W. Chang, D. Stehlé, and G. Villard. Perturbation analysis of the QR factor R in the context of LLL lattice basis reduction. 2011. To appear in Mathematics of Computation, available at `http://perso.ens-lyon.fr/damien.stehle/QRPERTURB.html`.

[21] D. Coppersmith and A. Shamir. Lattice attacks on NTRU. In *Proceedings of Eurocrypt*, volume 1233 of *LNCS*, pages 52–61. Springer, 1997.

[22] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9:251–280, 1990.

[23] J. Detrey, G. Hanrot, X. Pujol, and D. Stehlé. Accelerating lattice reduction with FPGAs. In *Proceedings of LATINCRYPT*, volume 6212 of *LNCS*, pages 124–143. Springer, 2010.

[24] I. Dinur, G. Kindler, and S. Safra. Approximating CVP to within almost polynomial factors is NP-hard. In *Proceedings of FOCS*, pages 99–109. IEEE Computer Society Press, 1998.

[25] F. Eisenbrand. Short vectors of planar lattices via continued fractions. *Inf. Process. Lett.*, 79(3):121–126, 2001.

[26] F. Eisenbrand. *50 Years of Integer Programming 1958-2008, From the Early Years to the State-of-the-Art*, chapter Integer Programming and Algorithmic Geometry of Numbers. Springer, 2009.

[27] F. Eisenbrand and G. Rote. Fast reduction of ternary quadratic forms. In *Proceedings of CALC*, volume 2146 of *LNCS*, pages 32–44. Springer, 2001.

[28] P. van Emde Boas. Another NP-complete partition problem and the complexity of computing short vectors in a lattice. Technical report 81-04, Mathematisch Instituut, Universiteit van Amsterdam, 1981.

[29] H. R. P. Ferguson and D. H. Bailey. A polynomial time, numerically stable integer relation algorithm. RNR Technical Report RNR-91-032; July 14, 1992.

[30] U. Fincke and M. Pohst. A procedure for determining algebraic integers of given norm. In *Proceedings of EUROCAL*, volume 162 of *LNCS*, pages 194–202, 1983.

[31] U. Fincke and M. Pohst. Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. *Math. Comp.*, 44(170):463–471, 1985.

[32] M. Fürer. Faster integer multiplication. *SIAM J. Comput*, 39(3):979–1005, 2009.

[33] N. Gama, N. Howgrave-Graham, H. Koy, and P. Q. Nguyen. Rankin's constant and blockwise lattice reduction. In *Proceedings of CRYPTO*, number 4117 in LNCS, pages 112–130. Springer, 2006.

[34] N. Gama and P. Q. Nguyen. Finding short lattice vectors within Mordell's inequality. In *Proceedings of STOC*, pages 207–216. ACM, 2008.

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

[36] N. Gama, P. Q. Nguyen, and O. Regev. Lattice enumeration using extreme pruning. In *Proceedings of Eurocrypt*, volume 6110 of *LNCS*, pages 257–278. Springer, 2010.

[37] J. von zur Gathen and J. Gerhardt. *Modern Computer Algebra, 2nd edition*. Cambridge University Press, 2003.

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

[39] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of STOC*, pages 197–206. ACM, 2008.

[40] O. Goldreich. *Foundations of Cryptography*, volume I – Basic Applications. Cambridge University Press, 2004.

[41] M. Gruber and C. G. Lekkerkerker. *Geometry of Numbers*. North-Holland, 1987.

[42] G. Hanrot, X. Pujol, and D. Stehlé. Algorithms for the shortest and closest lattice vector problems. In *Proceedings of IWCC*, volume 6639 of *LNCS*, pages 159–190. Springer, 2011.

[43] G. Hanrot, X. Pujol, and D. Stehlé. Analyzing blockwise lattice algorithms using dynamical systems, 2011. To appear in the proceedings of CRYPTO. Full version available at `http://perso.ens-lyon.fr/damien.stehle/BKZ.html`.

[44] G. Hanrot and D. Stehlé. Improved analysis of Kannan's shortest lattice vector algorithm (extended abstract). In *Proceedings of CRYPTO*, volume 4622 of *LNCS*, pages 170–186. Springer, 2007. Extended version available at `http://perso.ens-lyon.fr/damien.stehle/KANNAN_EXTENDED.html`.

[45] G. Hanrot and D. Stehlé. Worst-case Hermite-Korkine-Zolotarev reduced lattice bases. *CoRR*, abs/0801.3331, 2008.

[46] G. Hanrot and D. Stehlé. A complete worst-case analysis of Kannan's shortest lattice vector algorithm, 2011. Work in progress. Available at `http://perso.ens-lyon.fr/damien.stehle`.

[47] I. Haviv and O. Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In *Proceedings of STOC*, pages 469–477. ACM, 2007.

[48] B. Helfrich. Algorithms to construct Minkowski reduced and Hermite reduced lattice bases. *Theor. Comput. Science*, 41:125–139, 1985.

[49] J. Hermans, M. Schneider, J. Buchmann, F. Vercauteren, and B. Preneel. Parallel shortest lattice vector enumeration on graphics cards. In *Proceedings of Africacrypt*, volume 6055 of *LNCS*, pages 52–68. Springer, 2010.

[50] N. Higham. *Accuracy and Stability of Numerical Algorithms, 2nd edition*. SIAM, 2002.

[51] M. van Hoeij. Factoring polynomials and 0-1 vectors. In *Proceedings of the 2001 Cryptography and Lattices Conference (CALC'01)*, volume 2146 of *LNCS*, pages 45–50. Springer, 2001.

[52] M. van Hoeij and A. Novocin. Gradual sub-lattice reduction and a new complexity for factoring polynomials. In *Proceedings of LATIN*, volume 6034 of *LNCS*, pages 539–553. Springer, 2010.

[53] J. Hoffstein, N. Howgrave-Graham, J. Pipher, and W. Whyte. Practical lattice-based cryptography: `NTRUEncrypt` and `ntrusign`, 2009. Chapter of [86].

[54] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: a new high speed public key cryptosystem. Preprint; presented at the rump session of Crypto'96, 1996.

[55] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: a ring based public key cryptosystem. In *Proceedings of ANTS*, volume 1423 of *LNCS*, pages 267–288. Springer, 1998.

[56] IEEE P1363. Standard specifications for public-key cryptography. `http://grouper.ieee.org/groups/1363/`.

[57] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *Proceedings of STOC*, pages 12–24. ACM, 1989.

[58] E. Kaltofen. On the complexity of finding short vectors in integer lattices. In *Proceedings of EUROCAL'83*, volume 162 of *LNCS*, pages 236–244. Springer, 1983.

[59] R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proceedings of STOC*, pages 99–108. ACM, 1983.

[60] R. Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, 1987.

[61] P. N. Klein. Finding the closest lattice vector when it's unusually close. In *Proceedings of SODA*, pages 937–941. ACM, 2000.

[62] D. Knuth. The analysis of algorithms. In *Actes du Congrès International des Mathématiciens de 1970*, volume 3, pages 269–274. Gauthiers-Villars, 1971.

[63] H. Koy and C. P. Schnorr. Segment LLL-reduction of lattice bases. In *Proceedings of the 2001 Cryptography and Lattices Conference (CALC'01)*, volume 2146 of *LNCS*, pages 67–80. Springer, 2001.

[64] D. H. Lehmer. Euclid's algorithm for large numbers. *American Mathematical Monthly*, 45:227–233, 1938.

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

[66] L. Lovász. *An Algorithmic Theory of Numbers, Graphs and Convexity*. SIAM, 1986. CBMS-NSF Regional Conference Series in Applied Mathematics.

[67] V. Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In *Proceedings of ASIACRYPT*, volume 5912 of *LNCS*, pages 598–616. Springer, 2009.

[68] V. Lyubashevsky and D. Micciancio. Generalized compact knapsacks are collision resistant. In *Proceedings ICALP (2)*, volume 4052 of *LNCS*, pages 144–155. Springer, 2006.

[69] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In *Proceedings of Eurocrypt*, volume 6110 of *LNCS*, pages 1–23. Springer, 2010.

[70] R.J. McEliece. A public-key cryptosystem based on algebraic number theory. Technical report, Jet Propulsion Laboratory, 1978. DSN Progress Report 42-44.

[71] D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Comput. Complexity*, 16(4):365–411, 2007.

[72] D. Micciancio and S. Goldwasser. *Complexity of lattice problems: a cryptographic perspective*. Kluwer Academic Press, 2002.

[73] D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput*, 37(1):267–302, 2007.

[74] D. Micciancio and O. Regev. Lattice-based cryptography. In *Post-Quantum Cryptography, D. J. Bernstein, J. Buchmann, E. Dahmen (Eds)*, pages 147–191. Springer, 2009.

[75] D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. `http://cseweb.ucsd.edu/~pvoulgar/pub.html`.

[76] 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. ACM, 2010.

[77] D. Micciancio and P. Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proceedings of SODA*. ACM, 2010. Implementation available at `http://cseweb.ucsd.edu/~pvoulgar/impl.html`.

[78] I. Morel, D. Stehlé, and G. Villard. H-LLL: using Householder inside LLL. In *Proceedings of ISSAC*, pages 271–278. ACM, 2009.

[79] T. Mulders and A. Storjohann. On lattice reduction for polynomial matrices. *Journal of Symbolic Computation*, 35(4):377–401, 2003.

[80] P. Q. Nguyen. Hermite's constant and lattice algorithms. Chapter of [86].

[81] P. Q. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97. In *Proceedings of CRYPTO*, volume 1666 of *LNCS*, pages 288–304. Springer, 1999.

[82] P. Q. Nguyen and D. Stehlé. Floating-point LLL revisited. In *Proceedings of Eurocrypt*, volume 3494 of *LNCS*, pages 215–233. Springer, 2005.

[83] P. Q. Nguyen and D. Stehlé. LLL on the average. In *Proceedings of ANTS*, LNCS, pages 238–256. Springer, 2006.

[84] P. Q. Nguyen and D. Stehlé. An LLL algorithm with quadratic complexity. *SIAM J. Comput*, 39(3):874–903, 2009.

[85] P. Q. Nguyen and D. Stehlé. Low-dimensional lattice basis reduction revisited. *ACM Transactions on Algorithms*, 5(4), 2009. Article 46.

[86] P. Q. Nguyen and B. Vallée (editors). *The LLL Algorithm: Survey and Applications*. Information Security and Cryptography. Springer, 2009.

[87] P. Q. Nguyen and T. Vidick. Sieve algorithms for the shortest vector problem are practical. *Journal of Mathematical Cryptology*, 2(2), 2008.

[88] A. Novocin, D. Stehlé, and G. Villard. An LLL-reduction algorithm with quasi-linear time complexity. In *Proceedings of STOC*, pages 403–412. ACM, 2011. Full version available at `http://perso.ens-lyon.fr/damien.stehle/L1.html`.

[89] V. Y. Pan. *Structured matrices and polynomials, unified superfast algorithms*. Springer-Verlag and Birkhäuser, 2001.

[90] S. Paulus. Lattice basis reduction in function fields. In *Proceedings of ANTS*, volume 1423 of *LNCS*, pages 567–575. Springer, 1998.

[91] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proceedings of STOC*, pages 333–342. ACM, 2009.

[92] C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *Proceedings of the 2006 Theory of Cryptography Conference (TCC)*, pages 145–166, 2006.

[93] C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *Proceedings TCC*, volume 3876 of *LNCS*, pages 145–166. Springer, 2006.

[94] R. A. Perlner and D. A. Cooper. Quantum resistant public key cryptography: a survey. In *Proceedings of IDtrust*, pages 85–93. ACM, 2009.

[95] X. Pujol and D. Stehlé. Rigorous and efficient short lattice vectors enumeration. In *Proceedings of ASIACRYPT*, volume 5350 of *LNCS*, pages 390–405. Springer, 2008.

[96] X. Pujol and D. Stehlé. Solving the shortest lattice vector problem in time $2^{2.465n}$. Cryptology ePrint Archive, 2009. Available at `http://eprint.iacr.org/2009/605`.

[97] O. Regev. Lecture notes of *lattices in computer science*, course taught at the Computer Science Tel Aviv University. `http://www.cs.tau.il/~odedr`.

[98] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of STOC*, pages 84–93. ACM, 2005.

[99] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.

[100] O. Regev. The learning with errors problem, 2010. Invited survey in CCC 2010, available at `http://www.cs.tau.ac.il/~odedr/`.

[101] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[102] C. P. Schnorr. Progress on LLL and lattice reduction. Chapter of [86].

[103] C. P. Schnorr. A hierarchy of polynomial lattice basis reduction algorithms. *Theor. Comput. Science*, 53:201–224, 1987.

[104] C. P. Schnorr. A more efficient algorithm for lattice basis reduction. *Journal of Algorithms*, 9(1):47–62, 1988.

[105] C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. In *Proceedings of FCT'91*, volume 529 of *LNCS*, pages 68–85. Springer, 1991.

[106] C. P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Mathematics of Programming*, 66:181–199, 1994.

[107] C. P. Schnorr and H. H. Hörner. Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In *Proceedings of Eurocrypt*, volume 921 of *LNCS*, pages 1–12. Springer, 1995.

[108] A. Schönhage. Schnelle Berechnung von Kettenbruchentwicklungen. *Acta Informatica*, 1:139–144, 1971.

[109] A. Schönhage. Factorization of univariate integer polynomials by Diophantine approximation and improved basis reduction algorithm. In *Proceedings of ICALP*, volume 172 of *LNCS*, pages 436–447. Springer, 1984.

[110] A. Schönhage. Fast reduction and composition of binary quadratic forms. In *Proceedings of ISSAC*, pages 128–133. ACM, 1991.

[111] A. Schönhage and V. Strassen. Schnelle Multiplikation grosser Zahlen. *Computing*, 7:281–292, 1971.

[112] P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of FOCS*, pages 124–134. IEEE Computer Society Press, 1994.

[113] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput*, 26(5):1484–1509, 1997.

[114] V. Shoup. NTL, Number Theory C++ Library. Available at `http://www.shoup.net/ntl/`.

[115] C. L. Siegel. *Lectures on the Geometry of Numbers*. Springer, 1989.

[116] D. Stehlé. On the randomness of bits generated by sufficiently smooth functions. In *Proceedings of ANTS*, volume 4076 of *LNCS*, pages 257–274. Springer, 2006.

[117] D. Stehlé, V. Lefèvre, and P. Zimmermann. Searching worst cases of a one-variable function. *IEEE Transactions on Computers*, 54(3):340–346, 2005.

[118] D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In *Proceedings of Eurocrypt*, volume 6632 of *LNCS*, pages 27–47. Springer, 2011.

[119] D. Stehlé, R. Steinfeld, K. Tanaka, and K. Xagawa. Efficient public key encryption based on ideal lattices. In *Proceedings of Asiacrypt*, volume 5912 of *LNCS*, pages 617–635. Springer, 2009.

[120] D. Stehlé and M. Watkins. On the extremality of an 80-dimensional lattice. In *Proceedings of ANTS*, volume 6197 of *LNCS*, pages 340–356. Springer, 2010.

[121] A. Storjohann. Faster algorithms for integer lattice basis reduction. Technical report, ETH Zürich, available at `ftp://ftp.inf.ethz.ch/pub/publications/tech-reports/2xx/249.ps.gz`, 1996.

[122] C. K. Yap. Fast unimodular reduction: planar integer lattices. In *Proceedings of FOCS*, pages 437–446. IEEE Computer Society Press, 1992.

[123] H. Zha. A componentwise perturbation analysis of the QR decomposition. *SIAM J. Matrix Anal. Appl.*, 14(4):1124–1131, 1993.

MATHEMATICS OF COMPUTATION
Volume 00, Number 0, Pages 000–000
S 0025-5718(XX)0000-0

PERTURBATION ANALYSIS OF THE QR FACTOR R IN THE CONTEXT OF LLL LATTICE BASIS REDUCTION

XIAO-WEN CHANG, DAMIEN STEHLÉ, AND GILLES VILLARD

Abstract. In 1982, Arjen Lenstra, Hendrik Lenstra Jr. and László Lovász introduced an efficiently computable notion of reduction of basis of a Euclidean lattice that is now commonly referred to as LLL-reduction. The precise definition involves the R-factor of the QR factorisation of the basis matrix. In order to circumvent the use of rational/exact arithmetic with large bit-sizes, it is tempting to consider using floating-point arithmetic with small precision to compute the R-factor. In the present article, we investigate the accuracy of the factor R of the QR factorisation of an LLL-reduced basis. Our main contribution is the first fully rigorous perturbation analysis of the R-factor of LLL-reduced matrices under column-wise perturbations. Our results are very useful to devise LLL-type algorithms relying on floating-point approximations.

1. Introduction

Let $B \in \mathbb{R}^{m\times n}$ be of a full column rank matrix. It has a unique QR factorization $B = QR$, where the Q-factor $Q \in \mathbb{R}^{m\times n}$ has orthonormal columns, i.e., $Q^TQ = I$ (where I is the identity matrix), and the R-factor $R \in \mathbb{R}^{n\times n}$ is upper triangular with positive diagonal entries (see, e.g., [6, §5]). This fundamental tool in matrix computations is central to the LLL reduction algorithm, named after the authors of [12], which aims at efficiently finding reduced bases of Euclidean lattices.

A Euclidean lattice L is a discrete subgroup of $\mathbb{R}^m$ and it can always be represented by a full column rank basis matrix $B \in \mathbb{R}^{m\times n}$: $L = \{B\mathbf{x}, \mathbf{x} \in \mathbb{Z}^n\}$. If $n \geq 2$, L has infinitely many bases. They are related by unimodular transforms, i.e., multiplication on the right of B by an $n \times n$ integer matrix with determinant ± 1. Given a lattice, one is often interested in obtaining a basis whose vectors are short and close to being orthogonal. Refining the quality of a basis is generically called lattice reduction. Among many others, lattice reduction has applications in cryptology [19], algorithmic number theory [4], communications [16], etc. LLL takes as input a basis matrix B and returns a basis of the same lattice which is made of vectors whose norm product is not arbitrarily larger than the lattice determinant $\det L = \sqrt{\det(B^TB)}$ (see Theorem 5.2). More informatively, LLL returns a new basis matrix of the same lattice whose jth basis vector has norm not arbitrarily larger than the norm of the orthogonal projection of this basis vector onto the

2000 *Mathematics Subject Classification.* Primary 11H06, 65F25; Secondary 11Y99, 65F35.
Key words and phrases. lattice reduction, LLL, QR factorization, perturbation analysis.
Xiao-Wen Chang's work was supported by NSERC of Canada Grant RGPIN217191-07.
Damien Stehlé's work was partly funded by the LaRedA ANR project.
Gilles Villard's work was partly funded by the Gecko ANR project.

orthogonal complement of the space spanned by the first $j-1$ basis vectors for each $j \geq 2$.

The original LLL algorithm [12] assumed that the input basis is integral and used integer arithmetic for the operations on the basis and rational arithmetic for the operations on the R-factor. The bit-size of each rational (the bit-size of a/b with $a, b \in \mathbb{Z}$ is the sum of the bit-sizes of a and b) is bounded by a polynomial in the bit-sizes of the input matrix entries. Nevertheless, the cost of the rational arithmetic grows quickly and dominates the overall cost. Schnorr [22] was the first to use approximations of these rationals in a rigorous way. His algorithm was improved recently by Nguyen and Stehlé [17, 18] who significantly decreased the bit-size required for each approximation, and thus the overall complexity of the LLL-reduction. (Note that contrarily to [17, 18] Schnorr's approximations are not relying on standard floating-point arithmetic.) To further decrease the required precision and therefore the cost, Schnorr [11, 23, 24] suggested using the Householder QR factorization algorithm instead of the Cholesky factorization algorithm as was used in [17, 18], since it is known that the R-factor computed by Householder's algorithm is more accurate than the one computed with the Cholesky factorization of $B^T B$.

The R-factor of the matrix B varies continuously with B. If we consider a perturbed matrix $B + \Delta B$ that is sufficiently close to B (note that in the perturbation matrix ΔB, Δ does not represent anything, i.e., ΔB is not a product of Δ and B), then its R-factor $R + \Delta R$ remains close to R. The goal of the present article is to investigate how ΔB affects ΔR, for LLL-reduced matrices B. This perturbation analysis helps understanding and providing (a priori) guarantees on the quality of numerically computed factors R. The QR-factorization is typically computed by Householder reflections, Givens rotations or the modified Gram-Schmidt orthogonalization. These algorithms are backward stable with respect to the R-factor: if the computations are performed in floating-point arithmetic, then the computed $\widehat{R}$ is the true R-factor of a matrix $\widehat{B}$ which is very close to the input matrix B (see [7, §18]). Along with the backward stability analysis, a perturbation analysis provides accuracy bounds on the computed $\widehat{R}$. In the present paper, we consider a perturbation ΔB that satisfies

$$|\Delta B| \leq \varepsilon C |B|, \tag{1.1}$$

where $c_{i,j} = 1$ for all i, j and $\varepsilon > 0$ is a small scalar (it will be specified in the relevant theorems to be given in the paper how small it needs to be for the results to hold). The motivation for considering such a class of perturbations is that the backward rounding error from a rounding error analysis of the standard QR factorization algorithms fits in this class with $\varepsilon = O(u)$, where we omitted the dependence with respect to the matrix dimensions and u is the unit roundoff (see [7, Th. 18.4] and Theorem 6.4 given later).[1]

Our results. Our main contribution is the first fully rigorous perturbation analysis of the R-factor of LLL-reduced matrices under the perturbation (1.1) (Theorem 5.6). In order to make this result consistent with the LLL-reduction (i.e., the

[1]Note that the description of the backward error in [7, Th. 18.4] was modified in the newer edition [8, Th. 19.4]. In the latter, the matrix equation (1.1) is replaced by $\|\Delta \mathbf{b}_i\| \leq \varepsilon \|\mathbf{b}_i\|$, for all i. The two formulations are equivalent (up to a small factor that is polynomial in the matrix dimensions), but the matrix equation (1.1) is more suited for our sensitivity analysis.

perturbed reduced basis remains reduced, possibly with respect to weaker reduction parameters), we introduce *a new notion of LLL-reduction* (Definition 5.3). Matrices reduced in this new sense satisfy essentially the same properties as those satisfied by matrices reduced in the classical sense. But the new notion of reduction is more natural with respect to column-wise perturbations, as the perturbation of a reduced basis remains reduced (this is not the case with the classical notion of reduction). Another important ingredient of the main result, that may be of independent interest, is the improvement of the perturbation analyses of [1] and [28] for general full column rank matrices (section 2). More precisely, all our bounds are fully rigorous, in the sense that no higher order error term is neglicted, and explicit constant factors are provided. Explicit and rigorous bounds are invaluable for *guaranteeing computational accuracy*: one can choose a precision that will be known in advance to provide a certain degree of accuracy in the result. In [1, §6], a rigorous error bound was proved. A (much) smaller bound was given in [1, §8], but it is a first-order bound, i.e., high-order terms were neglected. Our rigorous bound is close to this improved bound. Our approach to deriving this rigorous bound is new and has been extended to the perturbation analysis of some other important matrix factorizations [3]. Finally, we give explicit constants in the backward stability analysis of Householder's algorithm from [8, §19], which, along with the perturbation analysis, provides fully rigorous and explicit error bounds for the computed R-factor of a LLL-reduced matrix.

Implications. Our results are descriptive in nature. However, the rigorous and explicit error analysis and the new notion of LLL-reducedness should lead to significant algorithmic improvements. Intuitively, we formalize the idea that only the $O(n)$ most significant bits of the vectors matter for their LLL-reducedness. Such a property has dramatic algorithmic consequences, as it implies that instead of computing with all bits we shall try to make use of only $O(n)$ bits for each matrix entry. For instance, in a context similar to [27], our result implies that in order to check the LLL-reducedness of a matrix, one only needs to consider $O(n)$ most significant bits of each column. This provides a $O(n^5)$-time (resp. $O(n^{4+\varepsilon})$-time) LLL certificate with naive integer arithmetic (resp. with FFT-based arithmetic [26]). Also, our results have been used to devise an efficient algorithm that improves the LLL-reducedness of an already LLL-reduced basis [15]. That algorithm finds a good unimodular transform by looking only at the $O(n)$ most significant bits of each column of the input matrix. Furthermore, the present work is the first step towards achieving Schnorr's goal of an LLL algorithm relying on the floating-point Householder algorithm. This goal has been reached in [14], which relies on the present results. Finally, these results helped devising an LLL-reduction algorithm whose bit-complexity is quasi-linear in fixed dimension [21], in the fashion of the Knuth-Schönhage quasi-linear time gcd algorithm [10, 25]. Roughly speaking, the first k bits of the quotients sequence of Euclid's gcd algorithm depends only on the first $2k$ bits of the two input integers. Knuth and Schönhage use that property to compute the quotients sequence by looking only at the first bits of the remainders sequence. Adapting this strategy to lattices involves truncations and hence perturbations of the basis vectors.

Road map. In section 2, we give our perturbation analysis of the R-factor for general full column matrices. Sections 3, 4 and 5 specialize the analysis to different

sets of matrices, including LLL-reduced matrices. Finally, in section 6, we provide explicit backward error bounds for Householder's QR factorization algorithm.

NOTATION. If $\mathbf{b}$ is a vector, then $\|\mathbf{b}\|_p$ denotes its ℓ_p norm. If $p = 2$, we omit the subscript. The jth column of a matrix $A = (a_{i,j})$ is denoted by $\mathbf{a}_j$ and $|A|$ denotes $(|a_{i,j}|)$. We use the MATLAB notation to denote submatrices: The matrix $A(i_1 : i_2, j_1 : j_2)$ consists of rows i_1 to i_2 and columns j_1 to j_2 of A; If i_1 and i_2 (resp. j_1 and j_2) are omitted, then all the rows (resp. columns) of A are kept; Finally, if $i_1 = i_2$ (resp. $j_1 = j_2$), we will write $A(i_1, j_1 : j_2)$ (resp. $A(i_1 : i_2, j_1)$). The Frobenius norm is $\|A\|_F = (\sum_{i,j} a_{i,j}^2)^{1/2}$. The ℓ_p matrix norm is $\|A\|_p = \sup_{\mathbf{x}\in\mathbb{R}^n} \|A\mathbf{x}\|_p/\|\mathbf{x}\|_p$. We use $\|A\|_{1,\infty}$ to denote either the 1-norm or the ∞-norm. We have $\|A\|_2 \le \|A\|_F$. If A and B are of compatible sizes, then $\|AB\|_F \le \|A\|_F\|B\|_2$ (see [8, Pbm. 6.5]) and $\|AB\|_2 \le \|A\|_2\|B\|_2$. If A is a square matrix, then $\mathrm{up}(A)$ denotes the upper triangular matrix whose ith diagonal entry is $a_{i,i}/2$ and whose upper-diagonal entries match those of A. We let $\mathcal{D}_n \subseteq \mathbb{R}^{n\times n}$ be the set of diagonal matrices with positive diagonal entries. For any nonsingular matrix X we define

$$\mathrm{cond}_2(X) = \left\| |X||X^{-1}| \right\|_2 . \tag{1.2}$$

If a is a real number, then $fl(a)$ denotes the floating-point number closest to a (with even mantissa when a is exactly half-way from two consecutive floating-point numbers). As a side-effect of our bounds being fully explicit, and since we tried to give tight and explicit perturbation bounds, some of theses bounds involve rather complicated and uninteresting terms. To make the presentation more compact, we encapsulate them in the variables $c_1, c_2, \ldots$.

2. Refined Perturbation Analysis of the R-Factor

In this section, we first give a general matrix-norm perturbation bound, then derive a column-wise perturbation bound.

2.1. A matrix-norm perturbation bound.

We will present a rigorous bound (i.e., without any implicit higher order term) on the perturbation of the R-factor when B is under the perturbation (1.1). In order to do that, we need the following two technical lemmas.

Lemma 2.1. *Let $n > 0$, $X \in \mathbb{R}^{n\times n}$ and $D = \mathrm{diag}(\delta_1, \ldots, \delta_n) \in \mathcal{D}_n$. We define $\zeta_D = 1$ for $n = 1$ and, for $n \ge 2$:*

$$\zeta_D = \sqrt{1 + \max_{1\le i<j\le n} (\delta_j/\delta_i)^2}. \tag{2.1}$$

Then we have

$$\|\mathrm{up}(X) + D^{-1}\mathrm{up}(X^T)D\|_F \le \zeta_D\|X\|_F, \tag{2.2}$$

and in particular, when $X^T = X$ and $D = I$,

$$\|\mathrm{up}(X)\|_F \le \frac{1}{\sqrt{2}}\|X\|_F. \tag{2.3}$$

Proof. The inequality (2.2) was given in [2, Lemma 5.1]. The inequality (2.3), which was given in [2, Eq. (2.3)], can also be derived from (2.2). □

The following provides a sufficient condition for the rank to be preserved during a continuous change from a full column-rank matrix B to $B + \Delta B$. This ensures that the R-factor is well-defined on the full path. This is of course not true if the matrix B is close to being rank deficient and the perturbation ΔB is not small, but that situation is prevented by assumption (2.4).

Lemma 2.2. *Let $B \in \mathbb{R}^{m\times n}$ be of full column rank with QR factorization $B = QR$. Let the perturbation matrix $\Delta B \in \mathbb{R}^{m\times n}$ satisfy (1.1). If*

$$\operatorname{cond}_2(R)\varepsilon < \frac{c}{m\sqrt{n}}, \tag{2.4}$$

for some constant $0 < c \le 1$, then the matrix $B + t\Delta B$ has full column rank for any $|t| \le 1$. Furthermore, $\|\Delta B R^{-1}\|_F < c$.

Proof. The second assertion follows from (2.4). In fact, from (1.1) and (2.4), we obtain

$$\begin{aligned}\|\Delta B R^{-1}\|_F &\le \varepsilon \big\| C|Q||R||R^{-1}| \big\|_F \le \varepsilon \big\|C\big\|_F \big\|Q\big\|_F \big\||R||R^{-1}|\big\|_2 \\ &= \varepsilon m\sqrt{n}\ \operatorname{cond}_2(R) < c.\end{aligned}$$

We now consider the first assertion. Notice that

$$Q^T(B + t\Delta B) = R + tQ^T\Delta B = \left(I + tQ^T\Delta B R^{-1}\right) R.$$

But $\|tQ^T\Delta B R^{-1}\|_2 \le \|\Delta B R^{-1}\|_2 < 1$, thus $I + tQ^T\Delta B R^{-1}$ is non-singular. So is $Q^T(B + t\Delta B)$, and hence $B + t\Delta B$ must have full column rank. □

Using the above two lemmas, we can prove the following perturbation theorem.

Theorem 2.3. *Let $B \in \mathbb{R}^{m\times n}$ be of full column rank with QR factorization $B = QR$. Let the perturbation matrix $\Delta B \in \mathbb{R}^{m\times n}$ satisfy (1.1). If*

$$\operatorname{cond}_2(R)\varepsilon < \frac{\sqrt{3/2} - 1}{m\sqrt{n}}, \tag{2.5}$$

then $B + \Delta B$ has a unique QR factorization

$$B + \Delta B = (Q + \Delta Q)(R + \Delta R), \tag{2.6}$$

and

$$\frac{\|\Delta R\|_F}{\|R\|_2} \le c_1(m,n)\chi(B)\,\varepsilon, \tag{2.7}$$

where, with ζ_D defined in (2.1)*:*

$$c_1(m,n) = (\sqrt{6} + \sqrt{3})mn^{1/2}, \tag{2.8}$$

$$\chi(B) = \inf_{D\in\mathcal{D}_n} \chi(R,D), \quad \chi(R,D) = \frac{\zeta_D \big\||R||R^{-1}|D\big\|_2 \big\|D^{-1}R\big\|_2}{\|R\|_2}. \tag{2.9}$$

Proof. The condition (2.5) ensures that (2.4) holds with $c = \sqrt{3/2} - 1$. Then, by Lemma 2.2, $B + t\Delta B$ is of full column rank for any $|t| \le 1$. Thus $B + t\Delta B$ has the unique QR factorization

$$B + t\Delta B = (Q + \Delta Q(t))(R + \Delta R(t)), \tag{2.10}$$

which, with $\Delta Q(1) = \Delta Q$ and $\Delta R(1) = \Delta R$, gives (2.6).

From (2.10), we obtain $(B + t\Delta B)^T(B + t\Delta B) = (R + \Delta R(t))^T(R + \Delta R(t))$, leading to

$$R^T\Delta R(t) + \Delta R(t)^T R = tR^TQ^T\Delta B + t\Delta B^TQR + t^2\Delta B^T\Delta B - \Delta R(t)^T\Delta R(t).$$

Multiplying the above by R^{-T} from the left and R^{-1} from the right, we obtain

$$\begin{aligned} &R^{-T}\Delta R(t)^T + \Delta R(t)R^{-1} \\ &= tQ^T\Delta BR^{-1} + tR^{-T}\Delta B^TQ + R^{-T}\left(t^2\Delta B^T\Delta B - \Delta R(t)^T\Delta R(t)\right)R^{-1}. \end{aligned}$$

Since $\Delta R(t)R^{-1}$ is upper triangular, it follows that

$$\begin{aligned} \Delta R(t)R^{-1} = \ & \mathrm{up}(tQ^T\Delta BR^{-1} + tR^{-T}\Delta B^TQ) \\ & + \mathrm{up}(t^2R^{-T}\Delta B^T\Delta BR^{-1}) - \mathrm{up}\big[R^{-T}\Delta R(t)^T\Delta R(t)R^{-1}\big]. \end{aligned} \tag{2.11}$$

Taking the F-norm on both sides of (2.11) and using Lemma 2.1 and the orthogonality of Q, we obtain

$$\|\Delta R(t)R^{-1}\|_F \le \sqrt{2}|t|\cdot\|\Delta BR^{-1}\|_F + \frac{1}{\sqrt{2}}t^2\|\Delta BR^{-1}\|_F^2 + \frac{1}{\sqrt{2}}\|\Delta R(t)R^{-1}\|_F^2. \tag{2.12}$$

Let $\rho(t) = \|\Delta R(t)R^{-1}\|_F$ and $\delta(t) = |t|\cdot\|\Delta BR^{-1}\|_F$. Then from (2.12)

$$\rho(t)(\sqrt{2} - \rho(t)) \le \delta(t)(2 + \delta(t)).$$

Here the left hand side has its maximum of $1/2$ with $\rho(t) = 1/\sqrt{2}$ and is increasing with respect to $\rho(t) \in [0, 1/\sqrt{2}]$. But, by Lemma 2.2, for $|t| \le 1$,

$$0 \le \delta(t) \le \|\Delta BR^{-1}\|_F < c = \sqrt{3/2} - 1. \tag{2.13}$$

This implies that $0 \le \delta(t)(2 + \delta(t)) < 1/2$ and $\rho(t)$, starting from 0, cannot reach its maximum. Hence $\rho(t) < 1/\sqrt{2}$ for any $|t| \le 1$. In particular, when $t = 1$,

$$\|\Delta RR^{-1}\|_F < 1/\sqrt{2}. \tag{2.14}$$

For any matrices $X \in \mathbb{R}^{n\times n}$ and $D \in \mathcal{D}_n$, we have $\mathrm{up}(XD) = \mathrm{up}(X)D$. Thus from (2.11) with $t = 1$ it follows that

$$\begin{aligned} \Delta RR^{-1}D = \ & \mathrm{up}\left[(Q^T\Delta BR^{-1}D) + D^{-1}(DR^{-T}\Delta B^TQ)D\right] \\ & + \mathrm{up}(R^{-T}\Delta B^T\Delta BR^{-1}D) - \mathrm{up}(R^{-T}\Delta R^T\Delta RR^{-1}D). \end{aligned} \tag{2.15}$$

Then, using Lemma 2.1, the inequality $\|\mathrm{up}(X)\|_F \le \|X\|_F$ for any $X \in \mathbb{R}^{n\times n}$ and the orthogonality of Q, we obtain from (2.15) that

$$\begin{aligned} \|\Delta RR^{-1}D\|_F \le \ & \zeta_D\|\Delta BR^{-1}D\|_F + \|\Delta BR^{-1}\|_F\|\Delta BR^{-1}D\|_F \\ & + \|\Delta RR^{-1}\|_F\|\Delta RR^{-1}D\|_F. \end{aligned}$$

Therefore, with (1.1), (2.13) and (2.14), we have

$$\begin{aligned} \|\Delta RR^{-1}D\|_F &\le \frac{\zeta_D + \sqrt{3/2} - 1}{1 - 1/\sqrt{2}}\|C\|_F\|Q\|_F\left\||R||R^{-1}|D\right\|_2 \\ &\le (\sqrt{6} + \sqrt{3})\zeta_D mn^{1/2}\left\||R||R^{-1}|D\right\|_2, \end{aligned}$$

where in deriving the second inequality we used the fact that $\zeta_D \ge 1$. Therefore,

$$\begin{aligned} \|\Delta R\|_F = \|\Delta RR^{-1}DD^{-1}R\|_F &\le \|\Delta RR^{-1}D\|_F\|D^{-1}R\|_2 \\ &\le (\sqrt{6} + \sqrt{3})\zeta_D mn^{1/2}\left\||R||R^{-1}|D\right\|_2\|D^{-1}R\|_2, \end{aligned}$$

leading to the bound (2.7). □

Remark 1. Theorem 2.3 is a rigorous version of a first-order perturbation bound given in [1, §8], which also involves $\chi(B)$. The new bound given here shows that if (2.4) holds then the high-order terms ignored in [1, §8] are indeed negligible. Numerical tests given in [1, §9] indicated that the first-order bound is a good approximation to the relative perturbation error in the R-factor. This suggests that the rigorous bound (2.7) is a good bound. By taking $D = I$ in (2.9), we obtain $\chi(B) \leq \sqrt{2}\mathrm{cond}_2(R)$. The quantity $\mathrm{cond}_2(R)$ is involved in the rigorous perturbation bound obtained in [1, §6] and can be arbitrarily larger than $\chi(B)$.

Remark 2. If the assumptions of Theorem 2.3 hold for B with perturbation ΔB, then they also hold for BS, for any arbitrary column scaling $S \in \mathcal{D}_n$, with perturbation ΔBS. The new R-factor is RS and the corresponding error is ΔRS. However, the quantity $\chi(B)$ is not preserved under column scaling.

2.2. **A column-wise perturbation bound.** For $j = 1, \ldots, n$, we define $R_j = R(1\!:\!j, 1\!:\!j)$, $\Delta R_j = \Delta R(1\!:\!j, 1\!:\!j)$, $\mathbf{r}_j = R(1\!:\!j, j)$ and $\Delta \mathbf{r}_j = \Delta R(1\!:\!j, j)$. Using Zha's approach [28, Cor. 2.2], we derive the following result.

Corollary 2.4. *If the assumptions of Theorem 2.3 hold, then for* $j = 1, \ldots, n$,

$$\frac{\|\Delta \mathbf{r}_j\|}{\|\mathbf{r}_j\|} \leq c_1(m, j)\chi(B, j)\varepsilon, \tag{2.16}$$

where

$$\chi(B, j) = \inf_{D \in \mathcal{D}_j} \chi(R, D, j) \geq 1, \quad \chi(R, D, j) = \frac{\zeta_D \big\| |R_j||R_j^{-1}|D \big\|_2 \|D^{-1}\mathbf{r}_j\|}{\|\mathbf{r}_j\|}. \tag{2.17}$$

Proof. For any $j \leq n$, we define $B_j = B(:, 1\!:\!j)$ and $\Delta B_j = \Delta B(:, 1\!:\!j)$. Note that $|\Delta B_j| \leq \varepsilon C|B_j|$ and $\mathrm{cond}_2(R_j)\varepsilon \leq \mathrm{cond}_2(R)\varepsilon \leq (\sqrt{3/2} - 1)/(m\sqrt{n})$. Thanks to Remark 2, we can apply Theorem 2.3 to $B_j S$ for an arbitrary $S \in \mathcal{D}_j$ with the perturbation matrix $\Delta B_j S$. Therefore, for any $D \in \mathcal{D}_j$,

$$\|\Delta R_j S\|_F \leq c_1(m, j)\zeta_D \big\| |R_j||R_j^{-1}|D \big\|_2 \big\|D^{-1}R_j S\big\|_2 \varepsilon.$$

Now, let the jth diagonal entry of S be 1 and the others tend to zero. Taking the limit provides (2.16). The lower bound on $\chi(B, j)$ in (2.17) follows from $\zeta_D \geq 1$ and

$$\big\| |R_j||R_j^{-1}|D \big\|_2 \|D^{-1}\mathbf{r}_j\| \geq \big\| |R_j||R_j^{-1}|DD^{-1}|\mathbf{r}_j| \big\| \geq \big\| |R_j R_j^{-1}\mathbf{r}_j| \big\| = \|\mathbf{r}_j\|.$$

□

Remark 3. The quantity $\chi(B, j)$ can be interpreted as an upper bound on the condition number of the jth column of R with respect to the perturbation ΔB of B. It is easy to check that the lower bound 1 on $\chi(B, j)$ in (2.17) is reached when $j = 1$, i.e., that $\chi(B, 1) = 1$.

In the following sections, we specialize Theorem 2.3 and Corollary 2.4 to several different classes of matrices, that are naturally linked to the LLL reduction.

3. Perturbation Analysis for Size-Reduced Matrices

We now study $\chi(B, j)$ for the class of size-reduced matrices, defined as follows.

Definition 3.1. Let $\eta \geq 0$. A full column-rank matrix $B \in \mathbb{R}^{m \times n}$ with R-factor R is η-size-reduced if for any $1 \leq i < j \leq n$, we have $|r_{i,j}| \leq \eta \cdot r_{i,i}$.

A matrix is 1-size-reduced if the largest element in magnitude in each row of the R-factor is reached on the diagonal. An example is the QR factorization with standard column pivoting (see, e.g., [6, Sec. 5.4.1]): one permutes the columns of the considered matrix so that for any $j \leq n$, the jth column is the one maximising $r_{j,j}$ among the last $n - j + 1$ columns. If column pivoting is used, then the sorted matrix is 1-size-reduced. The LLL algorithm [12] has a sub-routine usually called size-reduction which aims at computing a 1/2-size-reduced matrix by multiplying the initial matrix on the right by an integer matrix whose determinant is equal to 1 or -1. In the L^2 algorithm from [18], a similar sub-routine, relying on floating-point arithmetic, aims at computing an η-size-reduced matrix, for any specified $\eta > 1/2$.

In subsection 3.1, we establish an upper bound on $\chi(B, j)$. That upper bound corresponds to a particular choice of scaling D in $\chi(R, D, j)$. In subsection 3.2, we compare our particular scaling with the different scalings discussed in [1, §9]. We then give a geometric interpretation of the result we obtain in subsection 3.3.

3.1. **Perturbation bounds for size-reduced matrices.** We first propose a way of selecting a good diagonal matrix D in (2.9) and in (2.17) to bound $\chi(B)$ and $\chi(B, j)$, respectively. Combined with Theorem 2.3 and Corollary 2.4, this directly provides matrix-norm and column-wise perturbation bounds.

Theorem 3.2. *Let $B \in \mathbb{R}^{m\times n}$ with full column rank be η-size-reduced and let R be its R-factor. For $j = 1, \ldots, n$, we define $r'_{j,j} = r_{j,j}/\max_{1\leq k\leq j} r_{k,k}$ and $D'_j = \mathrm{diag}(r'_{1,1}, \ldots, r'_{j,j})$. Then*

$$\chi(B) \leq 2(1 + (n-1)\eta)(1+\eta)^{n-1}\zeta_{D'_n}, \tag{3.1}$$

$$\chi(B, j) \leq c_2(j, \eta)(1+\eta)^j \zeta_{D'_j}\big(\max_{1\leq k\leq j} r_{k,k}\big)/\|\mathbf{r}_j\|, \quad j = 1, \ldots, n, \tag{3.2}$$

where ζ_D is defined in (2.1) *for any arbitrary positive diagonal matrix D, and*

$$c_2(j, \eta) = 2\sqrt{1 + (j-1)\eta^2}/(1+\eta). \tag{3.3}$$

Proof. Let R'_j be obtained from R_j by dividing the kth column by $\max_{1\leq i\leq k} r_{i,i}$, for $k = 1, \ldots, j$. The diagonal entries of R'_j match $r'_{i,i}$'s from D'_j. Since R_j is η-size-reduced, so is R'_j. Let $T_j = {D'_j}^{-1} R'_j$. We have $t_{i,i} = 1$ and $t_{i,k} \leq \eta$ for $k > i$. Therefore, we have $|T_j^{-1}| \leq U_j^{-1}$, where $U_j \in \mathbb{R}^{j\times j}$ is upper triangular with $u_{i,i} = 1$ and $u_{i,k} = -\eta$ for $k > i$, see, e.g., [8, Th. 8.12]. Since $V_j = U_j^{-1}$ satisfies $v_{i,i} = 1$ and $v_{i,k} = \eta(1+\eta)^{k-i-1}$ for $k > i$ (see, e.g., [8, Eq. (8.4)]), we obtain

$$\begin{aligned} |R_j||R_j^{-1}|D'_j &= |R'_j||{R'_j}^{-1}|D'_j = D'_j|T_j||T_j^{-1}| \\ &\leq D'_j|U_j||V_j| = D'_j \begin{bmatrix} 1 & 2\eta & 2\eta(1+\eta) & \cdot & 2\eta(1+\eta)^{j-2} \\ & 1 & 2\eta & \cdot & 2\eta(1+\eta)^{j-3} \\ & & \cdot & \cdot & \cdot \\ & & & 1 & 2\eta \\ & & & & 1 \end{bmatrix}. \end{aligned}$$

Since $|r'_{i,i}| \leq 1$ for any i, we have

$$\big\||R_j||R_j^{-1}|D'_j\big\|_{1,\infty} \leq \big(1 + 2\eta\sum_{k=0}^{j-2}(1+\eta)^k\big) \leq 2(1+\eta)^{j-1}. \tag{3.4}$$

Notice that $|r_{p,q}|/r'_{p,p} = |r_{p,q}| \max_{1\le k\le p} r_{k,k}/r_{p,p} \le \eta \max_{1\le k\le p} r_{k,k}$. It follows that $|{D'_j}^{-1} R_j| \le (\max_{1\le k\le j} r_{k,k})|U_j|$. Therefore,

$$\|{D'_j}^{-1} R_j\|_{1,\infty} \le (1+(j-1)\eta) \max_{1\le k\le j} r_{k,k}, \quad \left\|{D'_j}^{-1}\mathbf{r}_j\right\| \le \sqrt{1+(j-1)\eta^2} \max_{1\le k\le j} r_{k,k}.$$

Then from the above and (3.4), and using the fact that $\|S\|_2 \le (\|S\|_1\|S\|_\infty)^{1/2}$ for any matrix S (see, e.g., [8, Eq. (6.19)]), we obtain

$$\frac{\left\||R||R^{-1}|D'_n\right\|_2 \|{D'_n}^{-1}R\|_2}{\|R\|_2} \le 2(1+(n-1)\eta)(1+\eta)^{n-1},$$
$$\frac{\left\||R_j||R_j^{-1}|D'_j\right\|_2 \left\|{D'_j}^{-1}\mathbf{r}_j\right\|}{\|\mathbf{r}_j\|} \le \frac{2\sqrt{1+(j-1)\eta^2}(1+\eta)^{j-1}\max_{1\le k\le j} r_{k,k}}{\|\mathbf{r}_j\|}.$$

Thus from (2.9) and (2.17) we conclude that (3.1) and (3.2) hold, respectively. □

Remark 4. Suppose we use the standard column pivoting strategy in computing the QR factorization of B. Then $r_{i,i} \ge r_{k,k}$ for $i < k \le j$, implying that $\zeta_{D'_j} \le \sqrt{2}$. Then, if P is the pivoting permutation matrix

$$\chi(BP) \le \sqrt{2}n2^n \quad \text{and} \quad \chi(BP, j) \le \sqrt{2j}2^j r_{1,1}/\|\mathbf{r}_j\|.$$

A similar bound on $\chi(BP)$ was given in [1, Th. 8.2].

3.2. **Choosing the row scaling in $\chi(R, D)$.** In [1, §9], Chang and Paige suggest different ways of choosing D in $\chi(R, D)$ to approximate $\chi(B)$. One way is to choose $D_r := \mathrm{diag}(\|R(i,:)\|)$ and $D = I$ and take $\min\{\chi(R, D_r), \chi(R, I)\}$ as an approximation to $\chi(B)$. The other way is to choose $D = D_e$ (see below for the definition of D_e) and use $\chi(R, D_e)$ as an approximation to $\chi(B)$.

The following matrix shows that the scaling D' from Theorem 3.2 can provide a much better approximation to $\chi(B)$ than $\min(\chi(R, D_r), \chi(R, I))$. Let

$$B = R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \gamma & \eta\gamma \\ 0 & 0 & 1/\gamma \end{bmatrix}.$$

When γ goes to infinity, both $\chi(R, D_r)$ and $\chi(R, I)$ tend to infinity, whereas $\chi(R, D')$ remains bounded. This also indicates that $\min\{\chi(R, D_r), \chi(R, I)\}$ can be significantly larger than $\chi(B)$.

The scaling D_e is constructed from $D_c R^{-1}$ with $D_c = \mathrm{diag}(\|\mathbf{r}_i\|_1)$. If we assume that B is a generic η-size-reduced matrix (or, more formally, that each $r_{i,j}$ is uniformly and independently distributed in $[-\eta \cdot r_{i,i}, \eta \cdot r_{i,i}]$), then with high probability D_c is the same as $\mathrm{diag}(\max_{1\le k\le i} r_{k,k})$, up to a polynomial factor in n. We have $D_c D^{-1} \le D_c|R^{-1}| \le D_c|V|D^{-1}$, where V is as in the proof of Theorem 3.2 and $D = \mathrm{diag}(r_{i,i})$. This implies that up to a factor exponential in n, $\|(D_c R^{-1})(:, i)\|$ is $1/r'_{i,i}$. The diagonal matrix D_e is defined by $D_e(i,i) = \min_{1\le k\le i} 1/\|(D_c R^{-1})(:,k)\|^2$. Up to factors exponential in n and for generic η-size-reduced matrices, the scaling D_e can be equivalently defined by $D_e(i,i) = \min_{1\le k\le i} r'_{k,k}$. A bound similar to the one of Theorem 3.2 can be derived for the latter scaling. Nevertheless, if R is diagonal, then $D_e = I$ and $\chi(R, D_e) = \sqrt{2}$, but $\chi(R, D')$ can be significantly larger. Finally, one may note that it is not known how

[2]The description of D_e in [1, §9] has an unintended error.

to compute D_e from R in $O(n^2)$ arithmetic operations or less, while computing D' requires only $O(n)$ arithmetic operations.

3.3. **Geometric interpretation of Theorem 3.2.** It is easy to verify that

$$\max_{1\le k\le i\le j}(r'_{i,i}/r'_{k,k}) \le \zeta_{D'_j} \le \sqrt{2}\max_{1\le k\le i\le j}(r'_{i,i}/r'_{k,k}).$$

When $(\max_{1\le i\le j} r_{i,i})/\|\mathbf{r}_j\| = O(1)$, e.g., for a generic η-size-reduced matrix with $|r_{i,j}|$ expected to be somewhat proportional to $r_{i,i}$, we see from (3.2) that the quantity $\max_{1\le k\le i\le j}(r'_{i,i}/r'_{k,k})$ bounds (up to a multiplicative factor that depends only on j) the sensitivity of the jth column of the R-factor. Let $x \mapsto r(x)$ be the piecewise affine interpolating function defined on $[1, n]$ such that $r(j) = r_{j,j}$ for $j = 1, \ldots, n$. For x_1 and x_2 in $[1, n]$ such that $r(x_1) = r(x_2)$, we consider the quantity $\max_{x\in[x_1,x_2]} r(x_1)/r(x) = \max_{x\in[x_1,x_2]} r(x_2)/r(x)$, which, as illustrated by Figure 1, represents the multiplicative depth of the graph of r between x_1 and x_2.

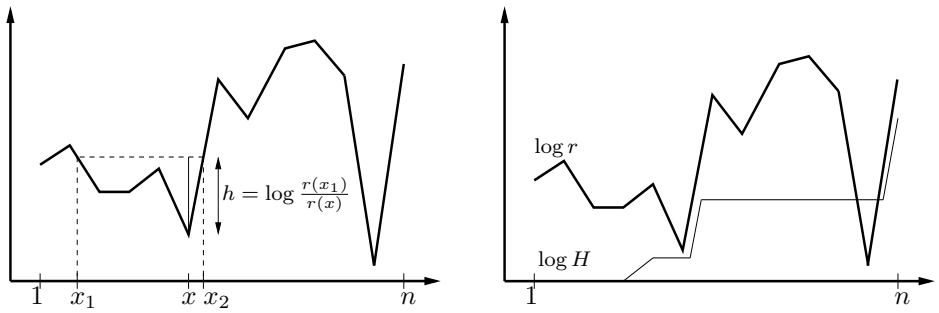

FIGURE 1. A possible graph of $\log r$: on the left hand side, with a depth h between x_1 and x_2 (the multiplicative depth is $\exp(h)$); on the right hand side, with the additive height function $\log H$.

We define the maximum depth before $r_{j,j}$ as:

$$H_j = \max_{1\le x_1\le x_2\le j,\ r(x_1)=r(x_2)} \left(\max_{x\in[x_1,x_2]} \frac{r(x_1)}{r(x)} \right),$$

which is illustrated on the right hand side of Figure 1. We now show the equivalence between $\zeta_{D'_j}$ and H_j. Without loss of generality, we consider only H_n.

Lemma 3.3. *We have* $H_n = \max_{1\le i\le j\le n}(r'_{j,j}/r'_{i,i})$.

Proof. We first prove that for any i and j such that $1 \le i \le j \le n$, $H_n \ge r'_{j,j}/r'_{i,i}$. We distinguish two cases, depending on the smallest index k_0 at which $\max_{1\le k\le j} r_{k,k}$ is reached. If $k_0 \le i$, then $r'_{j,j}/r'_{i,i} = r_{j,j}/r_{i,i}$. If $r_{j,j} \le r_{i,i}$, the result holds since $H_n \ge 1$; otherwise, we have $r_{j,j} > r_{i,i}$, leading to $H_n \ge r_{j,j}/r_{i,i}$ (in the definition of H_n, consider $x = i$, $x_2 = j$ and $x_1 \in [k_0, i]$ such that $r(x_1) = r(x_2)$). Suppose now that $i < k_0$. Since $r'_{j,j} \le 1$, we have $r'_{j,j}/r'_{i,i} \le \max_{1\le k\le i} r_{k,k}/r_{i,i}$. The latter is not greater than H_n (in the definition of H_n, consider $x = i$, $x_1 \le i$ such that $r(x_1) = \max_{1\le k\le i} r_{k,k}$ and $x_2 \in [i, k_0]$ such that $r(x_2) = r(x_1)$).

We now prove that $\max_{1\le i\le j\le n}(r'_{j,j}/r'_{i,i}) \ge H_n$. Let $x_1 \le x \le x_2$ in $[1, n]$ be such that $H_n = r(x_1)/r(x) = r(x_2)/r(x)$. We suppose that $x_1 < x < x_2$ as otherwise $H_n = 1 \le \max_{1\le i\le j\le n}(r'_{j,j}/r'_{i,i})$. By the definition of $r(\cdot)$, the real x

must be an integer. Similarly, either x_1 or x_2 is an integer. We consider these two cases separately. Suppose first that $x_1 \in \mathbb{Z}$. Then $r(x_1) \leq r(\lceil x_2 \rceil)$. We must have $\max_{1\leq k\leq \lfloor x_2 \rfloor} r_{k,k} = r_{x_1,x_1}$ and $\max_{1\leq k\leq \lceil x_2 \rceil} r_{k,k} = r_{\lceil x_2 \rceil,\lceil x_2 \rceil}$. This gives that $r'_{x,x} = r_{x,x}/r_{x_1,x_1}$ and $r'_{\lceil x_2 \rceil,\lceil x_2 \rceil} = 1$. Thus $H_n = r_{x_1,x_1}/r_{x,x} = r'_{\lceil x_2 \rceil,\lceil x_2 \rceil}/r'_{x,x}$. Suppose now that $x_2 \in \mathbb{Z}$. Then $r(\lfloor x_1 \rfloor) \geq r(x_2)$. Since $\max_{1\leq k\leq x_2} r_{k,k}$ is reached before x_1, we have $r'_{x_2,x_2}/r'_{x,x} = r_{x_2,x_2}/r_{x,x} = H_n$. □

If $(\max_{i\leq j} r_{i,i})/\|\mathbf{r}_j\| = O(1)$, then from Corollary 2.4 and Theorem 3.2 it follows that $\|\Delta\mathbf{r}_j\|/\|\mathbf{r}_j\| \lesssim (1+\eta)^j H_j \varepsilon$.

4. Perturbation Analysis for Weakly-Size-Reduced Matrices

The perturbation bounds given in Theorem 3.2 does not indicate that size-reducedness, as defined in section 3, is preserved after η-size-reduced B is perturbed to $B + \Delta B$. In fact, from (2.16) and (3.2),

$$|\Delta r_{i,j}| \leq \|\Delta\mathbf{r}_j\| \leq c_1(m,j)c_2(j,\eta)(1+\eta)^j \zeta_{D'_j}\Big(\max_{1\leq k\leq j} r_{k,k}\Big)\varepsilon, \tag{4.1}$$

and in particular,

$$|\Delta r_{i,i}| \leq c_1(m,i)c_2(i,\eta)(1+\eta)^i \zeta_{D'_i}\Big(\max_{1\leq k\leq i} r_{k,k}\Big)\varepsilon. \tag{4.2}$$

If the $r_{k,k}$'s are increasing, then the upper bound in (4.1) with $i < j$ can be arbitrarily larger than the upper bound in (4.2). Thus we cannot ensure that $|r_{i,j} + \Delta r_{i,j}| \leq \eta|r_{i,i} + \Delta r_{i,i}|$. Suppose we restrict ourselves to setting ε as a function of n only. Computationally, this corresponds to allowing ourselves to use arbitrary precision arithmetic, but with a precision that shall depend only on the dimension and not on the matrix entries. Then for any $\eta' > \eta$, one may choose $r_{k,k}$'s so that the perturbed basis cannot be guaranteed η'-size-reduced by the perturbation bound. Overall, this means that given a basis that we are told is η-size-reduced, and given $\eta' > \eta$, we cannot always ensure that it is η'-size-reduced, without setting the precision as a function of the matrix entries. This is a very undesirable computational property. For this reason, we modify the notion of size-reducedness. We will not be able to show that this new definition is preserved under the perturbation analysis of the R-factor (although the counter-example above will not work anymore): to obtain such a property, we will need a LLL-type set of conditions relying on the weakened size-reduction (see section 5).

Definition 4.1. Let $\eta, \theta \geq 0$. A full column-rank matrix $B \in \mathbb{R}^{m\times n}$ with R-factor R is (η,θ)-weakly-size-reduced ((η,θ)-WSR for short) if $|r_{i,j}| \leq \eta r_{i,i} + \theta r_{j,j}$ for any $i < j$.

The following matrix illustrates Definition 4.1: the magnitude of the coefficient $r_{i,j}$ is bounded with respect to both $r_{i,i}$ and $r_{j,j}$.

$$\begin{bmatrix} \ddots & & & & \\ & r_{i,i} & \xleftarrow{\ \eta\ } & r_{i,j} & \\ & & \ddots & \downarrow\theta & \\ & & & r_{j,j} & \\ & & & & \ddots \end{bmatrix}$$

As in the previous section, we analyze the quantity $\chi(B,j)$ by looking at the diagonal elements of the R-factor, i.e., the sequence of $r_{i,i}$'s.

Theorem 4.2. *Let $B \in \mathbb{R}^{m\times n}$ with full column rank be (η,θ)-WSR for some $\eta \geq 0$ and $\theta \geq 0$. Let R be its R-factor. For $j = 1,\ldots,n$, we let $r'_{j,j} = r_{j,j}/\max_{1\leq k\leq j} r_{k,k}$, $D'_j = \mathrm{diag}(r'_{1,1},\ldots,r'_{j,j})$ and $\xi_{D'_j} = \prod_{1\leq k<j} \max\left(\frac{r'_{k+1,k+1}}{r'_{k,k}}, 1\right)$. Then*

$$\chi(B,j) \leq \sqrt{2}c_2(j,\eta+\theta)(1+\eta+\theta)^j \xi_{D'_j}\big(\max_{1\leq k\leq j} r_{k,k}\big)/\|\mathbf{r}_j\|, \quad j = 1,\ldots,n. \tag{4.3}$$

Proof. Without loss of generality, we assume that $r_{1,1} = \max_{1\leq k\leq n} r_{k,k}$. If this is not the case, we divide the jth column of R by $\max_{1\leq k\leq j} r_{k,k}$ for $j = 1,\ldots,n$. Note that $\chi(B,j)$ is column-scaling invariant (see (2.17)), and that the quantities $(\max_{1\leq k\leq j} r_{k,k})/\|\mathbf{r}_j\|$ and $\xi_{D'_j}$ are invariant under this particular scaling.

Let $D = \mathrm{diag}(\xi_{D'_1},\ldots,\xi_{D'_n})$ and let $\bar{R} = RD^{-1}$. As $\chi(B,j)$ is invariant under column-scaling, we have $\chi(B,j) = \chi(BD^{-1},j)$. The most important part of the proof is to show that $\bar{R}$ is $\bar{\eta}$-size-reduced with $\bar{\eta} = \eta+\theta$. Once this is established, we will apply Theorem 3.2 to $\bar{R}$ to derive (4.3).

We want to prove that for any $i < j$, we have $|\bar{r}_{i,j}| \leq \bar{\eta}\bar{r}_{i,i}$. Because of the (η,θ)-WSR assumption, this will hold if

$$\eta\frac{r_{i,i}}{\xi_{D'_j}} + \theta\frac{r_{j,j}}{\xi_{D'_j}} \leq (\eta+\theta)\frac{r_{i,i}}{\xi_{D'_i}}.$$

Since $\xi_{D'_j} \geq \xi_{D'_i}$ when $j \geq i$, it suffices to prove that $\frac{r_{j,j}}{\xi_{D'_j}} \leq \frac{r_{i,i}}{\xi_{D'_i}}$, or equivalently that the sequence of the $\bar{r}_{i,i}$'s is non-increasing. This is equivalent to showing that $\frac{r_{j,j}}{\xi_{D'_j}} \leq \frac{r_{j-1,j-1}}{\xi_{D'_{j-1}}}$ holds for any $j \geq 2$, which is a direct consequence of the definition of $\xi_{D'_j}$.

We now apply Theorem 3.2 to BD^{-1}. For any $1 \leq j \leq n$, we have

$$\chi(B,j) = \chi(BD^{-1},j) \leq c_2(j,\bar{\eta})(1+\bar{\eta})^j \zeta_{\bar{D}'_j}\big(\max_{1\leq k\leq j} \bar{r}_{k,k}\big)/\|\bar{\mathbf{r}}_j\|,$$

with $\bar{D}'_j = \mathrm{diag}\left(\frac{\bar{r}_{i,i}}{\max_{1\leq k\leq i}\bar{r}_{k,k}}\right)_{1\leq i\leq j}$. The fact that the sequence of the $\bar{r}_{i,i}$'s is non-increasing implies that $\bar{D}'_j = \mathrm{diag}\left(\frac{\bar{r}_{i,i}}{\bar{r}_{1,1}}\right)_{1\leq i\leq j}$. For the same reason, we have $\zeta_{\bar{D}'_j} \leq \sqrt{2}$. This also gives that $\max_{1\leq k\leq j}\bar{r}_{k,k} = r_{1,1}$. Finally, we have $\|\bar{\mathbf{r}}_j\| = \|\mathbf{r}_j\|/\xi_{D'_j} = \|\mathbf{r}_j\|/\xi_{D_j}$. Since we assumed that $r_{1,1} = \max_{1\leq k\leq n} r_{k,k}$, this completes the proof. □

Remark 5. Naturally, as the assumption on B in Theorem 4.2 is weaker than in Theorem 3.2, the bound obtained for $\chi(B,j)$ is weaker as well. Indeed, it is easy to show that we always have $\zeta_{D'_j} \leq \sqrt{2}\xi_{D'_j}$. Furthermore, $\xi_{D'_j}$ can be arbitrarily larger than $\zeta_{D'_j}$. For instance, consider $\{r_{i,i}\}_{1\leq i\leq 5}$ defined by $r_{1,1} = r_{3,3} = r_{5,5} = 1$ and $r_{2,2} = r_{4,4} = \varepsilon$, where $\varepsilon > 0$ tends to 0. In this case, $\zeta_{D'_j} = O(1/\varepsilon)$, whereas $\xi_{D'_j} = O(1/\varepsilon^2)$.

Remark 6. Similarly to size-reduced matrices, we cannot argue from the perturbation results given in Corollary 2.4 and Theorem 4.2 that the weak size-reducedness is preserved after the perturbation (cf. the discussion given at the beginning of section 4). However, LLL-reduced matrices, which rely on weak size-reduction and will be introduced in section 5, do not have this drawback.

5. LLL Reduction is a Fix-Point Under Column-Wise Perturbation

In the present section, after some reminders on Euclidean lattices, we will introduce a modification of the LLL reduction [12] which is compatible with the perturbation analysis of the R-factor that we performed in the previous sections.

5.1. **Background on Euclidean lattices.** We give below the background on lattices that is necessary to the upcoming discussion. For more details, we refer to [13]. A *Euclidean lattice* is the set of all integer linear combinations of the columns of a full column rank basis matrix $B \in \mathbb{R}^{m\times n}$: $L = \{B\mathbf{x}, \mathbf{x} \in \mathbb{Z}^n\}$. The matrix B is said to be a basis matrix of L and its columns are a basis of L. If $n \geq 2$, a given lattice has infinitely lattice bases, but they are related to one another by *unimodular transforms*, i.e., by right-multiplication by $n\times n$ integer matrices of determinant ± 1. A lattice invariant is a quantity that does not depend on the particular choice of a basis of a given lattice. The simplest such invariant is the *lattice dimension* n. Let R be the R-factor of the basis matrix B. The *determinant* of the lattice L is defined as the product of the diagonal entries of R: $\det(L) = \prod_{1\leq i\leq n} r_{i,i}$. Since lattice bases are related by unimodular matrices, the determinant is a lattice invariant. Another important invariant is the *minimum* $\lambda(L)$ defined as the norm of a shortest non-zero vector of L.

Lattice reduction is a major paradigm in the theory of Euclidean lattices. The aim is to find a basis of good quality of a lattice given by an arbitrary basis. One usually targets orthogonality and norm properties. A simple reason why one is interested in short vectors is that they require less space to store. One is interested in basis matrices whose columns are fairly orthogonal relatively to their norms (which can be achieved by requiring the off-diagonal $r_{i,j}$'s to be small and the sequence of the $r_{i,i}$'s to not decrease too fast), for several different reasons. For example, it is crucial to bound the complexity of enumeration-type algorithms that find shortest lattice vectors and closest lattice vectors to given targets in the space [9, 5]. In 1982, Lenstra, Lenstra and Lovász [12] described a notion of reduction, called *LLL reduction*, that can be reached in time polynomial in the size of the input basis and that ensures some orthogonality and norm properties. Their algorithm immediately had great impact on various fields of mathematics and computer science (we refer to [20] for an overview).

Definition 5.1. Let $\eta \in [1/2, 1)$ and $\delta \in (\eta^2, 1]$. Let B be a lattice basis matrix and R be its R-factor. The basis matrix B is (δ, η)-LLL-reduced if it is η-size-reduced and if for any i we have $\delta \cdot r_{i,i}^2 \leq r_{i,i+1}^2 + r_{i+1,i+1}^2$.

Originally in [12], the parameter η was set to $1/2$, but this condition was relaxed later by Schnorr [22] to allow inaccuracies in the computation of the entries of the matrix R. Allowing $\eta > 1/2$ does not change significantly the guaranteed quality of LLL-reduced matrices (see below). The parameter δ was chosen to be $3/4$ in [12], because this simplifies the expressions of the constants appearing in the quality bounds of $(\delta, 1/2)$-LLL-reduced matrices (the α in Theorem 5.2 becomes $\sqrt{2}$). The second condition in Definition 5.1 means that after projection onto the orthogonal complement of the first $i-1$ columns, the ith one is approximately shorter (i.e., not much longer) than the $(i+1)$th. Together, the two conditions imply that the $r_{i,i}$'s cannot decrease too quickly and that the norm of the ith column is essentially $r_{i,i}$ (up to a factor that depends only of the dimension). The theorem below gives the main properties of LLL-reduced matrices.

Theorem 5.2. *Let $\eta \in [1/2, 1)$ and $\delta \in (\eta^2, 1]$. Let $\alpha = \frac{1}{\sqrt{\delta - \eta^2}}$. If $B \in \mathbb{R}^{m \times n}$ is a (δ, η)-LLL-reduced basis matrix of a lattice L, then we have:*

$$\begin{aligned} r_{j,j} &\leq \alpha \cdot r_{j+1,j+1}, \quad j = 1, \ldots, n-1, \\ \|\mathbf{b}_j\| &\leq \alpha^{j-1} \cdot r_{j,j}, \quad j = 1, \ldots, n, \\ \|\mathbf{b}_1\| &\leq \alpha^{n-1} \cdot \lambda(L), \\ \|\mathbf{b}_1\| &\leq \alpha^{\frac{n-1}{2}} \cdot (\det(L))^{\frac{1}{n}}, \\ \prod_{1 \leq j \leq n} \|\mathbf{b}_j\| &\leq \alpha^{\frac{n(n-1)}{2}} \cdot \det(L). \end{aligned}$$

We do not give a proof, since Theorem 5.2 is a simple corollary of Theorem 5.4.

5.2. **A weakening of the LLL-reduction.** LLL-reduction suffers from the same drawback as size-reduction with respect to column-wise perturbations. If the ε parameter of a column-wise perturbation is set as a function of n, then for any $\eta' > \eta$ and any $\delta' < \delta$, one may choose $r_{k,k}$'s so that the initial basis is (δ, η)-LLL-reduced but the perturbed basis cannot be guaranteed (δ', η')-size-reduced. Indeed, consider the matrix $\begin{bmatrix} 1 & 0 \\ 0 & \gamma \end{bmatrix}$, where γ grows to infinity. We can choose $\Delta r_{1,1} = 0$ and $\Delta r_{1,2} = \varepsilon\gamma$. The latter grows linearly with γ and eventually becomes bigger than any fixed η', thus preventing the perturbed matrix from being size-reduced.

For this reason, we introduce a weakening of LLL-reduction that relies on weak-size-reduction instead of size-reduction. This seems to be more coherent with the approximate computation of the R-factor of the QR factorization by Householder reflections, Givens rotations or the Modified Gram-Schmidt orthogonalization. The weakening has the nice property that if a basis is reduced according to this definition and the corresponding R-factor is computed by any of these algorithms using floating-point arithmetic, then it suffices to show that the basis is indeed reduced according to this weakening (up to a small additional relaxation of the same type). This relaxation is thus somehow a fix-point with respect to floating-point computation of the R-factor by these algorithms. We will make this statement precise in Corollary 6.5. The need for such a weakening was discovered by Schnorr [23, 24], though he did not define it formally nor proved any quality property.

Definition 5.3. Let $\eta \in [1/2, 1), \theta \geq 0$ and $\delta \in (\eta^2, 1]$. Let B be a lattice basis matrix and R be its R-factor. The basis matrix B is (δ, η, θ)-LLL-reduced if it is (η, θ)-WSR and if for any i we have: $\delta \cdot r_{i,i}^2 \leq r_{i,i+1}^2 + r_{i+1,i+1}^2$.

Figure 2 illustrates the different definitions of LLL-reduction. If the $r_{i,i}$'s are decreasing, then a (δ, η, θ)-LLL-reduced basis matrix is $(\delta, \eta + \theta)$-reduced. The weakening becomes more interesting when the $r_{i,i}$'s do not decrease. In any case, it does not worsen significantly the bounds of Theorem 5.2.

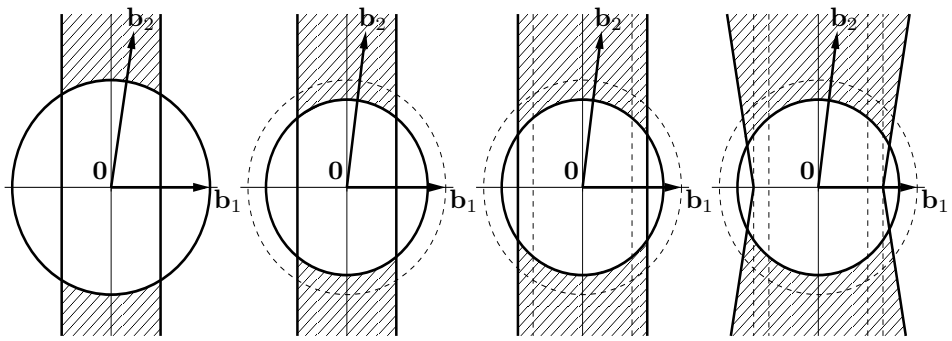

FIGURE 2. The hashed area is the set of vectors $\mathbf{b}_2$ such that $(\mathbf{b}_1, \mathbf{b}_2)$ is (from left to right) $(1,0,0)$-LLL, $(\delta,0,0)$-LLL, $(\delta,\eta,0)$-LLL and (δ,η,θ)-LLL.

Theorem 5.4. *Let $\eta \in [1/2, 1), \theta \geq 0$ and $\delta \in (\eta^2, 1]$. Let $\alpha = \frac{\theta\eta + \sqrt{(1+\theta^2)\delta - \eta^2}}{\delta - \eta^2}$. If $B \in \mathbb{R}^{m\times n}$ is a (δ,η,θ)-LLL-reduced basis matrix of a lattice L, then we have*

$$r_{j,j} \leq \alpha \cdot r_{j+1,j+1}, \quad j = 1, \ldots, n-1, \tag{5.1}$$

$$\|\mathbf{b}_j\| \leq \alpha^{j-1} \cdot r_{j,j}, \quad j = 1, \ldots, n, \tag{5.2}$$

$$\|\mathbf{b}_1\| \leq \alpha^{n-1} \cdot \lambda(L), \tag{5.3}$$

$$\|\mathbf{b}_1\| \leq \alpha^{\frac{n-1}{2}} \cdot (\det(L))^{\frac{1}{n}}, \tag{5.4}$$

$$\prod_{1\leq j\leq n} \|\mathbf{b}_j\| \leq \alpha^{\frac{n(n-1)}{2}} \cdot \det(L). \tag{5.5}$$

Here α is always greater than or equal to $\frac{1}{\sqrt{\delta-\eta^2}}$, the value of α defined in Theorem 5.2. However, when θ tends to 0, the former tends to the latter.

Proof. By the given conditions, we have:

$$\delta r_{j,j}^2 \leq (\eta r_{j,j} + \theta r_{j+1,j+1})^2 + r_{j+1,j+1}^2 \leq \eta^2 r_{j,j}^2 + 2\eta\theta r_{j,j} r_{j+1,j+1} + (1+\theta^2) r_{j+1,j+1}^2.$$

This implies that $x := \frac{r_{j,j}}{r_{j+1,j+1}}$ satisfies the following degree-2 inequality:

$$(\delta - \eta^2)x^2 - 2\eta\theta x - (1+\theta^2) \leq 0. \tag{5.6}$$

The discriminant is $4\left((1+\theta^2)\delta - \eta^2\right) > 0$ and the leading coefficient is non-negative. As a consequence, we have:

$$x \leq \frac{\theta\eta + \sqrt{(1+\theta^2)\delta - \eta^2}}{\delta - \eta^2} = \alpha,$$

leading to (5.1).

Now we show (5.2). From (5.6), we have $(\delta-\eta^2)\alpha^2 - 2\eta\theta\alpha - (1+\theta^2) = 0$. But $\delta \leq 1$. Thus $(1-\eta^2)\alpha^2 - 2\eta\theta\alpha - (1+\theta^2) \geq 0$, or equivalently $(\theta+\eta\alpha)^2 \leq \alpha^2 - 1$.

Using this fact and $\alpha \geq 1$ as well, we have

$$\begin{aligned}\|\mathbf{b}_j\|^2 = \sum_{1\leq i\leq j} r_{i,j}^2 &\leq r_{j,j}^2 + \sum_{1\leq i<j} \left(\eta^2 \cdot r_{i,i}^2 + 2\theta\eta \cdot r_{j,j} r_{i,i} + \theta^2 \cdot r_{j,j}^2\right) \\ &\leq \left(1 + \sum_{1\leq i<j} (\eta^2\alpha^{2(j-i)} + 2\theta\eta\alpha^{j-i} + \theta^2)\right) \cdot r_{j,j}^2 \\ &\leq \left(1 + \sum_{1\leq i<j} (\eta^2\alpha^2 + 2\theta\eta\alpha + \theta^2)\alpha^{2(j-i-1)}\right) \cdot r_{j,j}^2 \\ &\leq \left(1 + (\theta+\eta\alpha)^2 \, \frac{\alpha^{2(j-1)} - 1}{\alpha^2 - 1}\right) \cdot r_{j,j}^2 \leq \alpha^{2(j-1)} \cdot r_{j,j}^2,\end{aligned}$$

leading to (5.2).

From (5.1), we have $r_{j,j} \geq \alpha^{1-j} \cdot r_{1,1}$. Suppose that $\mathbf{z} \in \mathbb{Z}^n$ satisfies $z_i \neq 0$ while $z_j = 0$ for $j = i+1, \ldots, n$. Then

$$\|B\mathbf{z}\| = \|R\mathbf{z}\| \geq |r_{i,i} z_i| \geq r_{i,i} \geq \alpha^{1-i} r_{1,1} = \alpha^{1-i}\|\mathbf{b}_1\|.$$

We thus have $\lambda(L) = \min_{\mathbf{z}\in\mathbb{Z}^n, \mathbf{z}\neq 0} \|B\mathbf{z}\| \geq \alpha^{1-n}\|\mathbf{b}_1\|$, which proves (5.3).

Since $\det(L) = \prod_{1\leq j\leq n} r_{j,j} \geq \prod_{1\leq j\leq n} (\alpha^{1-j} \cdot r_{1,1}) = \alpha^{(n-1)n/2}\|\mathbf{b}_1\|^n$, (5.4) holds. The inequality (5.5) follows from (5.2). □

5.3. **Application to LLL-reduced matrices.** We first show that the assumption of Theorem 2.3 is fulfilled for (δ, η, θ)-reduced basis matrices. To do this, we bound $\mathrm{cond}_2(R)$ for any upper triangular basis matrix R which is reduced.

Lemma 5.5. *Let $\eta, \theta \geq 0$ and $\alpha \geq 1$. Suppose an upper triangular matrix $R \in \mathbb{R}^{n\times n}$ with positive diagonal entries satisfies*

$$|r_{i,j}| \leq \eta r_{i,i} + \theta r_{j,j}, \quad r_{i,i} \leq \alpha r_{i+1,i+1}, \quad j = i+1, \ldots, n, \quad i = 1, \ldots, n-1. \tag{5.7}$$

Then

$$\mathrm{cond}_2(R) \leq \frac{|1-\eta-\theta|\alpha + 1}{(1+\eta+\theta)\alpha - 1}(1+\eta+\theta)^n \alpha^n. \tag{5.8}$$

Proof. In the proof, we will use the following fact a few times: for any strictly upper triangular matrix $U \in \mathbb{R}^{n\times n}$, we have $(I-U)^{-1} = \sum_{0\leq k<n} U^k$.

Write $R = \bar{R} \cdot D$, where $D = \mathrm{diag}(r_{1,1}, \ldots, r_{n,n})$ and $\bar{r}_{i,j} = \frac{r_{i,j}}{r_{j,j}}$ for $i \leq j$. From the assumption (5.7) it follows that $|\bar{r}_{i,j}| \leq (\eta\alpha^{j-i} + \theta)$ for $i < j$. Define T to be the strictly upper triangular matrix with $t_{i,j} = \bar{r}_{i,j}$ for $i < j$. Let J be the matrix whose all entries are 0, except that all $(i, i+1)$ entries are 1's. The matrix T is nilpotent and satisfies

$$|T| \leq (\eta+\theta) \sum_{1\leq k<n} (\alpha J)^k = (\eta+\theta)\alpha J(I - \alpha J)^{-1}.$$

Since $\bar{R} = I + T$, we have

$$|\bar{R}| \leq I + (\eta+\theta)\alpha J(I-\alpha J)^{-1} = (I - (1-\eta-\theta)\alpha J)(I-\alpha J)^{-1}.$$

Since T is strictly upper triangular, $\bar{R}^{-1} = \sum_{0\leq k<n}(-T)^k$. As a consequence,

$$\begin{aligned}|\bar{R}^{-1}| \leq \sum_{0\leq k<n} |T|^k &\leq \sum_{0\leq k<n} \left[(\eta+\theta)\alpha J(I-\alpha J)^{-1}\right]^k \\ &= \left[I-(\eta+\theta)\alpha J(I-\alpha J)^{-1}\right]^{-1} \\ &= (I-\alpha J)\left(I-(1+\eta+\theta)\alpha J\right)^{-1} \\ &= (I-\alpha J)\sum_{0\leq k<n}(1+\eta+\theta)^k\alpha^k J^k.\end{aligned}$$

using the fact that $\|J\|_2 = 1$, we obtain

$$\begin{aligned}\left\||\bar{R}|\cdot|\bar{R}^{-1}|\right\|_2 &\leq \|I-(1-\eta-\theta)\alpha J\|_2 \sum_{0\leq k<n}\left\|(1+\eta+\theta)^k\alpha^k J^k\right\|_2 \\ &\leq (|1-\eta-\theta|\alpha+1)\sum_{0\leq k<n}(1+\eta+\theta)^k\alpha^k \\ &\leq \frac{|1-\eta-\theta|\alpha+1}{(1+\eta+\theta)\alpha-1}(1+\eta+\theta)^n\alpha^n.\end{aligned}$$

Using the equality $\mathrm{cond}_2(R) = \mathrm{cond}_2(\bar{R})$ allows us to assert that (5.8) holds. □

Remark 7. Let R be the upper triangular matrix with $r_{i,i} = \alpha^{-i}$ and $r_{i,j} = \eta\alpha^{-i}(-1)^{i-j+1}$ for $i < j$. Then R satisfies (5.7) with $\theta = 0$, and we have $\mathrm{cond}_2(R) \geq \eta\alpha^{n-1}(1+\eta)^{n-2}$, which is very close to the upper bound (5.8) with $\theta = 0$. Indeed, if we use the same notations as in the proof of Lemma 5.5, we have $\mathrm{cond}_2(R) = \mathrm{cond}_2(\bar{R})$ with $\bar{R} = I + \eta\alpha J - \eta\alpha^2 J^2 + \ldots = I + \eta\alpha J(I+\alpha J)^{-1}$. Then $\bar{R}^{-1} = I - \eta\alpha J(I+(1+\eta)\alpha J)^{-1}$. The proof is completed by noting that $\mathrm{cond}_2(\bar{R})$ is not smaller than the $(1,n)$-entry of $|\bar{R}|\cdot|\bar{R}^{-1}|$, which itself is not smaller than $\eta\alpha^{n-1}(1+\eta)^{n-2}$.

We now specialize our perturbation analysis of the previous sections to the case of (δ,η,θ)-LLL-reduced basis matrices.

Theorem 5.6. *Let $\eta \in [1/2, 1)$, $\theta \geq 0$, $\delta \in (\eta^2, 1]$ and $\alpha = \frac{\theta\eta+\sqrt{(1+\theta^2)\delta-\eta^2}}{\delta-\eta^2}$. Let $B \in \mathbb{R}^{m\times n}$ be a (δ,η,θ)-LLL-reduced basis matrix and R be its R-factor. Let $\Delta B \in \mathbb{R}^{m\times n}$ be a perturbation matrix satisfying* (1.1)*, where ε satisfies*

$$c_3(1+\eta+\theta)^n\alpha^n\varepsilon < 1, \tag{5.9}$$

with

$$c_3 = \frac{(|1-\eta-\theta|\alpha+1)m\sqrt{n}}{((1+\eta+\theta)\alpha-1)(\sqrt{3/2}-1)}. \tag{5.10}$$

Then $B+\Delta B$ has a unique R-factor $R+\Delta R$ and

$$\|\Delta\mathbf{r}_j\| \leq \sqrt{2}c_1(m,j)c_2(j,\eta+\theta)(1+\eta+\theta)^j\alpha^{k_j}r_{j,j}\varepsilon, \quad j = 1,\ldots,n, \tag{5.11}$$

where c_1 and c_2 are defined by (2.8) *and* (3.3)*, respectively, and k_j is the number of indices i such that $i < j$ and $r_{i,i} > r_{i+1,i+1}$.*

Proof. From Lemma 5.5, we see that the condition (5.9) ensures that the assumption (2.5) in Theorem 2.3 is satisfied. From Corollary 2.4 and Theorem 4.2 it follows that

$$\|\Delta \mathbf{r}_j\| \leq \sqrt{2} c_1(m,j) c_2(j,\eta+\theta) \xi_{D'_j} (1+\eta+\theta)^j \Big(\max_{1\leq i\leq j} r_{i,i} \Big) \varepsilon, \quad j = 1, \ldots, n, \tag{5.12}$$

where $\xi_{D'_j} = \prod_{i=1}^{j-1} \max\left(r'_{i+1,i+1}/r'_{i,i}, 1\right)$ with $r'_{j,j} = r_{j,j} / \max_{1\leq i\leq j} r_{i,i}$. If $r_{i,i} > r_{i+1,i+1}$ holds, then with (5.1) we have $r'_{i,i}/r'_{i+1,i+1} = r_{i,i}/r_{i+1,i+1} \leq \alpha$, thus $1 \leq \alpha \cdot r'_{i+1,i+1}/r'_{i,i}$. Then it follows that

$$\xi_{D'_j} = \left(\prod_{\substack{i=1 \\ r'_{i+1,i+1} \geq r'_{i,i}}}^{j-1} \frac{r'_{i+1,i+1}}{r'_{i,i}} \right) \cdot \left(\prod_{\substack{i=1 \\ r'_{i,i} > r'_{i+1,i+1}}}^{j-1} \alpha \frac{r'_{i+1,i+1}}{r'_{i,i}} \right) \leq \alpha^{k_j} \frac{r'_{j,j}}{r'_{1,1}} = \alpha^{k_j} r'_{j,j},$$

which, combined with (5.12), results in (5.11). □

Remark 8. It is also possible to obtain an upper bound on $\frac{\|\Delta \mathbf{r}_j\|}{r_{j,j}}$ by using (5.2), Corollary 2.4 with $D = I$, and Lemma 5.5. This allows to circumvent the more tedious analysis corresponding to sections 3 and 4. However, the bound obtained in this way is (much) larger.

We can now conclude that the set of LLL-reduced matrices is a fix-point under column-wise perturbations.

Corollary 5.7. *Let $\eta \in [1/2, 1)$, $\theta \geq 0$, $\delta \in (\eta^2, 1]$ and $\alpha = \frac{\theta\eta + \sqrt{(1+\theta^2)\delta - \eta^2}}{\delta - \eta^2}$. Let $B \in \mathbb{R}^{m\times n}$ be a (δ, η, θ)-LLL-reduced basis matrix. Let $\Delta B \in \mathbb{R}^{m\times n}$ be a perturbation matrix satisfying (1.1), where ε is such that*

$$\varepsilon' := c_4 (1+\eta+\theta)^n \alpha^n \varepsilon < 1,$$

with

$$c_4 = \max(c_3, \sqrt{2} c_1(m,n) c_2(n, \eta+\theta)), \tag{5.13}$$

and with c_1, c_2 and c_3 defined by (2.8), (3.3) and (5.10), respectively. Then $B + \Delta B$ is $(\delta', \eta', \theta')$-LLL-reduced with

$$\delta' = \delta \frac{(1-\varepsilon')^2}{(1+\varepsilon')^2 (1 + 2\varepsilon'(\eta\alpha + \theta))}, \ \eta' = \frac{\eta}{1-\varepsilon'} \ \text{ and } \ \theta' = \frac{\theta + \varepsilon'}{1-\varepsilon'}.$$

Proof. Let $R' = R + \Delta R$ be the R-factor of $B + \Delta B$. From Theorem 5.6, it follows that for all $1 \leq i \leq j \leq n$, we have $|\Delta r_{i,j}| \leq \varepsilon' r_{j,j}$. Therefore,

$$(1-\varepsilon') r_{i,i} \leq r'_{i,i} \leq (1+\varepsilon') r_{i,i} \ \text{ and } \ |r'_{i,j}| \leq \eta r_{i,i} + (\theta + \varepsilon') r_{j,j}.$$

As a consequence, we have $|r'_{i,j}| \leq \frac{\eta}{1-\varepsilon'} r'_{i,i} + \frac{\theta+\varepsilon'}{1-\varepsilon'} r'_{j,j}$, which gives us the weak-size-reduction. We also have

$$\begin{aligned} |r'_{i,i+1}| &\geq |r_{i,i+1}| - \varepsilon' r_{i+1,i+1} \\ (r'_{i,i+1})^2 &\geq r_{i,i+1}^2 - 2\varepsilon' |r_{i,i+1}| r_{i+1,i+1} \\ &\geq r_{i,i+1}^2 - 2\varepsilon' \left(\eta r_{i,i} + \theta r_{i+1,i+1}\right) r_{i+1,i+1} \\ &\geq r_{i,i+1}^2 - 2\varepsilon'(\eta\alpha + \theta) r_{i+1,i+1}^2. \end{aligned}$$

Therefore:

$$\frac{\delta}{(1+\varepsilon')^2}\cdot(r'_{i,i})^2 \le r^2_{i+1,i+1}+r^2_{i,i+1} \le r^2_{i+1,i+1}+(r'_{i,i+1})^2+2\varepsilon'(\eta\alpha+\theta)r^2_{i+1,i+1}$$
$$\le \frac{1+2\varepsilon'(\eta\alpha+\theta)}{(1-\varepsilon')^2}\left((r'_{i+1,i+1})^2+(r'_{i,i+1})^2\right).$$

This completes the proof. □

If the initial parameters δ, η and θ are such that $\eta \in (1/2, 1)$, $\theta > 0$, and $\delta \in (\eta^2, 1)$, then ε can be chosen as a function of δ, η, θ, m and n so that the resulting parameters δ', η', θ' also satisfy the domain conditions $\eta' \in (1/2, 1)$, $\theta' > 0$ and $\delta' \in ((\eta')^2, 1)$. Overall, this means that the set of basis matrices that are (δ, η, θ)-LLL-reduced for some parameters $\eta \in (1/2, 1)$, $\theta > 0$, and $\delta \in (\eta^2, 1)$ is stable under column-wise perturbations when ε is limited to a function of the parameters and the dimensions m and n only. Note that if we fix $\theta = 0$, we cannot guarantee that the perturbed basis is reduced with $\theta' = 0$. This is why the weakened LLL-reduction is more appropriate with respect to column-wise perturbations.

6. Practical Computation

In many cases, the perturbation matrix considered in a perturbation analysis comes from a backward stability result on some algorithm. In the case of QR factorization, the algorithms for which backward stability is established are the Householder algorithm, the Givens algorithm and the Modified Gram-Schmidt algorithm [8, §19]. In this section, we give a precise backward stability result for Householder's algorithm. We then apply it to LLL reduced bases. Similar results hold for the Givens and Modified Gram-Schmidt algorithms.

6.1. Backward stability of Householder's algorithm.

Columnwise error analysis of the Householder QR factorization algorithm has been given in [8, §19]. But the constant in the backward error bound is not precisely computed. However, this information is crucial for some applications, such as the LLL reduction, since it will allow one to select floating-point precision to provide correctness guarantees. The purpose of the present section is to give a precise backward error bound. The model of floating-point arithmetic that we use is formally described in [8, Eq. (2.4)].

Suppose we are given an $m \times n$ matrix B that has full column rank and that we aim at computing its R-factor R. Householder's algorithm proceeds column-wise by transforming B to R. Suppose that after j steps we have transformed B into a matrix of the following form:

$$\left(\begin{array}{c|c} B'_{1,1} & B'_{1,2} \\ \hline 0 & B'_{2,2} \end{array}\right),$$

where $B'_{1,1}$ is an $j \times j$ upper triangular matrix with positive diagonal entries. In the $(j+1)$th step, we apply a Householder transformation Q_{j+1} (which is orthogonal) to $B'_{2,2}$ from the left such that the first column of $B'_{2,2}$ becomes $[\times, 0, \ldots, 0]^T$. For the computation of the Householder transformation, see Figure 3, which gives two variants and is taken from [8, Lemma 19.1] with some changes. The Householder algorithm computes the full form of the QR factorization: $B = Q\begin{bmatrix} R \\ 0 \end{bmatrix}$, where $Q \in \mathbb{R}^{m\times m}$ is orthogonal and $R \in \mathbb{R}^{n\times n}$ is upper triangular. Some of the diagonal

entries of R may be negative, but if we want them to be positive, we can multiply the corresponding rows of R and columns of Q by -1.

Input: A vector $\mathbf{b} \in \mathbb{R}^m$.
Output: A vector $\mathbf{v} \in \mathbb{R}^m$ such that $Q = I - \mathbf{v}\mathbf{v}^T$ is orthogonal and $Q \cdot \mathbf{b} = (\pm\|\mathbf{b}\|, 0, \ldots, 0)^T$.

1. $\mathbf{v}$:=$\mathbf{b}$.
2. s:=sign$(b_1) \cdot \|\mathbf{b}\|$.
3. v_1:=$b_1 + s$ (variant A) or v_1:=$\frac{-\sum_{i=2}^m b_i^2}{b_1+s}$ (variant B).
4. $\mathbf{v}$:=$\frac{1}{\sqrt{s \cdot v_1}} \cdot \mathbf{v}$ (variant A) or $\mathbf{v}$:=$\frac{1}{\sqrt{-s \cdot v_1}} \cdot \mathbf{v}$ (variant B).

FIGURE 3. Two variants of computing the Householder transformation.

The algorithm of Figure 3 is performed with floating-point arithmetic. The computational details are straightforward, except for Step 3 of variant B: the numerator is a term that appears in the computation of Step 2, and thus does not need being re-computed. In our rounding error analysis, all given numbers are assumed to be real numbers (so they may not be floating-point numbers), and all algorithms are assumed to be run with unit roundoff u, i.e., $u = 2^{-p}$, where p is the precision. We use a hat to denote a computed quantity. For convenience, we use δ to denote a quantity satisfying $|\delta| \leq u$. The quantity $\gamma_m := \frac{mu}{1-mu}$ will be used a few times. The computations of some bounds contained in the proofs of the following lemmas were performed by MAPLE. The corresponding MAPLE work-sheet is available at `http://perso.ens-lyon.fr/damien.stehle/RPERTURB.html`.

The following lemma is a modified version of [8, Lemma 19.1].

Lemma 6.1. *Suppose we run either variant of the algorithm of Figure 3 on a nonzero vector $\mathbf{b} \in \mathbb{R}^m$ with unit roundoff u satisfying $c_5 \cdot u \leq 1$, where:*

$$c_5 = 4(6m + 63) \text{ for variant A, and } c_5 = 8(6m + 39) \text{ for variant B.} \tag{6.1}$$

Let $\hat{\mathbf{v}}$ be the computed vector and $\mathbf{v}$ be the vector that would have been computed with infinite precision. Then $\hat{\mathbf{v}} = \mathbf{v} + \Delta\mathbf{v}$ with $|\Delta\mathbf{v}| \leq (m+11)u \cdot |\mathbf{v}|$ for variant A (resp. $|\Delta\mathbf{v}| \leq \frac{1}{2}(5m + 29)u \cdot |\mathbf{v}|$ for variant B).

Proof. Let $c = \mathbf{b}^T\mathbf{b}$. Then $\hat{c} = fl(\hat{\mathbf{b}}^T\hat{\mathbf{b}})$ where $|\hat{\mathbf{b}} - \mathbf{b}| \leq u|\mathbf{b}|$. By following [8, p. 63], it is easy to verify that

$$\frac{|\hat{c} - c|}{|c|} \leq \gamma_{m+2}. \tag{6.2}$$

Note that the above result is different from [8, Eq. (3.5)], since here the b_i's are not assumed to be floating-point numbers. Since $\gamma_{m+2} < 1$, using (6.2) we have

$$\frac{|\sqrt{\hat{c}} - \sqrt{c}|}{\sqrt{c}} = \frac{|\hat{c} - c|}{\sqrt{c}} \frac{1}{\sqrt{\hat{c}} + \sqrt{c}} \leq \frac{|\hat{c} - c|}{2c\sqrt{1 - \gamma_{m+2}}} \leq \frac{\gamma_{m+2}}{2\sqrt{1 - \gamma_{m+2}}} =: \beta_1. \tag{6.3}$$

Then it follows that at Step 2,

$$\frac{|\hat{s} - s|}{|s|} = \frac{|\sqrt{\hat{c}}(1 + \delta) - \sqrt{c}|}{\sqrt{c}} \leq (1 + \beta_1)(1 + u) - 1 =: \beta_2. \tag{6.4}$$

We now consider variants A and B of the algorithm separately. For variant A and at Step 3, the quantities b_1 and s have the same sign, so $|b_1| + |s| = |b_1 + s|$. Thus, using (6.4) we have

$$\begin{aligned} \frac{|\hat{v}_1 - v_1|}{|v_1|} &= \frac{|(\hat{b}_1 + \hat{s})(1+\delta) - (b_1 + s)|}{|b_1 + s|} \le \frac{|\hat{b}_1(1+\delta) - b_1| + |\hat{s}(1+\delta) - s|}{|b_1 + s|} \\ &\le \frac{|b_1|[(1+u)^2 - 1] + |s|[(1+\beta_2)(1+u) - 1]}{|b_1 + s|} \le (1+\beta_2)(1+u) - 1 =: \beta_3, \end{aligned} \tag{6.5}$$

Then, using (6.4) and (6.5) we have

$$\frac{|\hat{d} - d|}{|d|} = \frac{|\hat{s}\hat{v}_1(1+\delta) - s v_1|}{|s v_1|} \le (1+\beta_2)(1+\beta_3)(1+u) - 1 =: \beta_4. \tag{6.6}$$

The MAPLE work-sheet shows that $\beta_4 < 1$. Let $e = \sqrt{d} = \sqrt{s v_1}$. Then, by the same derivation for (6.4) (see (6.3)), using (6.6) we have

$$\frac{|\hat{e} - e|}{|e|} = \frac{|\sqrt{\hat{d}}\,(1+\delta) - \sqrt{d}|}{\sqrt{d}} \le \left(1 + \frac{\beta_4}{2\sqrt{1-\beta_4}}\right)(1+u) - 1 := \beta_5. \tag{6.7}$$

The MAPLE work-sheet shows that $\beta_5 < 1$. Then from (6.5) and (6.7) we obtain the following componentwise bound:

$$|\hat{\mathbf{v}} - \mathbf{v}| \le \left(\frac{1+\beta_3}{1-\beta_5}(1+u) - 1\right)|\mathbf{v}| = \beta_6 |\mathbf{v}|, \tag{6.8}$$

where $\beta_6 = \frac{1+\beta_3}{1-\beta_5}(1+u) - 1 \le (m+11)u$, as indicated by the MAPLE work-sheet.

Now we consider variant B. The quantity $\sum_{i=2}^{m} b_i^2$ from Step 3 has been computed at Step 2. The relative error in the computed value is bounded by γ_{m+1}. Thus, using this fact and (6.5) (for the denominator) we conclude that

$$\frac{|\hat{v}_1 - v_1|}{|v_1|} \le \frac{1+\gamma_{m+1}}{1-\beta_3}(1+u) - 1 =: \beta_3'. \tag{6.9}$$

According to the MAPLE work-sheet, we have $\beta_3' < 1$. The rest analysis is similar to the derivation for (6.6)–(6.8) and we have the following componentwise bound:

$$|\hat{\mathbf{v}} - \mathbf{v}| \le \left(\frac{1+\beta_3'}{1-\beta_5'}(1+u) - 1\right)|\mathbf{v}| = \beta_6' |\mathbf{v}|, \tag{6.10}$$

where $\beta_5' = \left(1 + \frac{\beta_4'}{2\sqrt{1-\beta_4'}}\right)(1+u) - 1$, $\beta_4' = (1+\beta_2)(1+\beta_3')(1+u) - 1$ and $\beta_6' := \frac{1+\beta_3'}{1-\beta_5'}(1+u) - 1$. The MAPLE work-sheet shows that $\beta_6' \le \frac{1}{2}(5m+29)u$. □

At step $j+1$ of the QR factorization, once the Householder vector $\mathbf{v}$ is computed, the Householder matrix is applied to all the remaining column vectors of the matrix $B_{2,2}'$. The following lemma, a modified version of [8, Lemma 19.2], provides a backward analysis for this step.

Lemma 6.2. *Suppose that the assumptions of Lemma 6.1 hold. Let* $\mathbf{c} \in \mathbb{R}^m$, $Q = I - \mathbf{v}\mathbf{v}^T$ *and* $\mathbf{y} = Q\mathbf{c} = \mathbf{c} - \mathbf{v}(\mathbf{v}^T\mathbf{c})$. *In computing* $\mathbf{y}$, *the computed Householder vector* $\hat{\mathbf{v}}$ *is used. Then there exists* $\Delta Q \in \mathbb{R}^{m\times m}$ *such that*

$$\hat{\mathbf{y}} = (Q + \Delta Q)\mathbf{c} \quad \textit{and} \quad \|\Delta Q\|_F \le \frac{1}{4}c_5 u. \tag{6.11}$$

Proof. The proofs for both variants of the algorithm of Figure 3 are the same, so we only consider variant A. Let $t = \mathbf{v}^T\mathbf{c}$. Then $\hat{t} = fl(\hat{\mathbf{v}}^T\hat{\mathbf{c}})$, where $|\hat{\mathbf{c}} - \mathbf{c}| \leq u|\mathbf{c}|$ and $|\hat{\mathbf{v}}-\mathbf{v}| \leq \beta_6|\mathbf{v}|$, by Lemma 6.1. Then by following the derivation of [8, Eq. (3.4)], we can show that

$$|\hat{t} - t| \leq [(1+\beta_6)(1+u)(1+\gamma_m) - 1]\,|\mathbf{v}|^T|\mathbf{c}| = \beta_7|\mathbf{v}|^T|\mathbf{c}|, \tag{6.12}$$

where $\beta_7 := (1+\beta_6)(1+u)(1+\gamma_m)-1$. Let $\mathbf{w} = \mathbf{v}(\mathbf{v}^T\mathbf{c}) = \mathbf{v}t$. Then $\hat{\mathbf{w}} = \hat{\mathbf{v}}\hat{t}(1+\delta)$. Using (6.8), (6.10) and (6.12) we obtain the following bound:

$$|\hat{\mathbf{w}} - \mathbf{w}| \leq ((1+\beta_6)(1+\beta_7)(1+u) - 1)\,|\mathbf{v}||\mathbf{v}|^T|\mathbf{c}| = \beta_8|\mathbf{v}||\mathbf{v}|^T|\mathbf{c}|,$$

where $\beta_8 = (1+\beta_6)(1+\beta_7)(1+u) - 1$. Then it follows that

$$|\hat{\mathbf{y}}-\mathbf{y}| = |fl(\hat{\mathbf{c}}-\hat{\mathbf{w}})-(\mathbf{c}-\mathbf{w})| \leq [(1+u)^2-1]|\mathbf{c}|+[(1+\beta_8)(1+u)-1]|\mathbf{v}||\mathbf{v}|^T|\mathbf{c}|. \tag{6.13}$$

Note that the Householder vector $\mathbf{v}$ satisfies $\|\mathbf{v}\| = \sqrt{2}$. Thus from (6.13) it follows that

$$\|\hat{\mathbf{y}} - \mathbf{y}\| \leq [(1+u)^2 - 1]\|\mathbf{c}\| + 2[(1+\beta_8)(1+u) - 1]\|\mathbf{c}\| = \beta_9\|\mathbf{c}\|,$$

where $\beta_9 = (1+u)^2 + 2(1+\beta_8)(1+u) - 3$. We can write $\hat{\mathbf{y}} = (Q + \Delta Q)\mathbf{c}$ with $\Delta Q = \frac{(\hat{\mathbf{y}}-\mathbf{y})\mathbf{c}^T}{\mathbf{c}^T\mathbf{c}}$. We have $\|\Delta Q\|_F = \frac{\|\hat{\mathbf{y}}-\mathbf{y}\|}{\|\mathbf{c}\|} \leq \beta_9$. In the MAPLE work-sheet, we see that $\beta_9 \leq \frac{1}{4}c_5u$, and thus (6.11) holds. □

The following lemma is a modified version of [8, Lemma 19.3]. It considers error analysis of a sequence of Householder matrices applied to a given matrix.

Lemma 6.3. *Let $B \in \mathbb{R}^{m\times n}$ and let $Q_i = I - \mathbf{v}_i\mathbf{v}_i^T$ for $i \leq n$ be a sequence of Householder matrices. We consider the sequence of transformations $B_{i+1} = Q_iB_i$, with $B_1 = B$. Suppose that these transformations are performed by using the computed Householder vectors $\hat{\mathbf{v}}_i$ with unit roundoff u. Let*

$$c_6 = \frac{1}{2}nc_5, \tag{6.14}$$

with c_5 defined by (6.1). If $c_6u \leq 1$, then the computed matrix $\widehat{B}_{n+1}$ satisfies

$$\widehat{B}_{n+1} = Q^T(B + \Delta B),$$

where $Q^T = Q_nQ_{n-1}\ldots Q_1$ and

$$\|\Delta\mathbf{b}_j\| \leq c_6u\|\mathbf{b}_j\|, \qquad j = 1,\ldots,n. \tag{6.15}$$

Proof. Let $\mathbf{b}_j^{(n+1)}$ be the jth column of B_{n+1}. From Lemma 6.2 it follows that there exist $\Delta Q_1,\ldots,\Delta Q_n \in \mathbb{R}^{m\times m}$ such that

$$\hat{\mathbf{b}}_j^{(n+1)} = (Q_n + \Delta Q_n)\ldots(Q_1 + \Delta Q_1)\mathbf{b}_j \;\text{ and }\; \|\Delta Q_i\|_F \leq \frac{1}{4}c_5u.$$

Write $Q^T + \Delta Q^T = (Q_n + \Delta Q_n)\ldots(Q_1 + \Delta Q_1)$. Then by [8, Lemma 3.7] we have

$$\|\Delta Q^T\|_F \leq \left(1 + \frac{1}{4}c_5u\right)^n - 1 \leq \frac{\frac{1}{4}c_5nu}{1 - \frac{1}{4}c_5nu} \leq c_6u.$$

Define $\Delta\mathbf{b}_j = Q\Delta Q^T\mathbf{b}_j$. Then

$$\hat{\mathbf{b}}_j^{(n+1)} = Q^T\mathbf{b}_j + \Delta Q^T\mathbf{b}_j = Q^T(\mathbf{b}_j + \Delta\mathbf{b}_j) \quad\text{and}\quad \|\Delta\mathbf{b}_j\| \leq c_6u\|\mathbf{b}_j\|.$$

□

We can now conclude with a version of [7, Th. 18.4] (or [8, Th. 19.4], in the newer edition), with generic constants replaced by explicit constants.

Theorem 6.4. *Let $\widehat{R}$ be the computed R-factor of the QR factorization of a given matrix $B \in \mathbb{R}^{m\times n}$ by the Householder algorithm, with unit roundoff u. If $c_6 u \leq 1$ with $c_6 = 2n(6m+63)$ for variant A and $4n(6m+39)$ for variant B, then there exists an orthogonal matrix $Q \in \mathbb{R}^{m\times m}$ such that*

$$B + \Delta B = Q \begin{bmatrix} \widehat{R} \\ 0 \end{bmatrix} \quad \text{and} \quad \|\Delta \mathbf{b}_j\| \leq c_6 u \|\mathbf{b}_j\|, \quad j = 1, \ldots, n.$$

The latter implies that $|\Delta B| \leq c_6 u C|B|$, where $c_{i,j} = 1$ for all i, j. The matrix Q is given explicitly by $Q^T = Q_n Q_{n-1} \ldots Q_1$, where Q_i is the Householder matrix corresponding to the exact application of the ith step of the Householder algorithm to $\widehat{B}_i$.

Proof. As a direct consequence of Lemma 6.3, we have:

$$B + \Delta B = Q \begin{bmatrix} \widehat{R} \\ 0 \end{bmatrix} \quad \text{and} \quad \|\Delta \mathbf{b}_j\| \leq c_6 u \|\mathbf{b}_j\|, \quad j = 1, \ldots, n,$$

with $Q = Q_1^T Q_2^T \ldots Q_n^T$. Then

$$|\Delta b_{i,j}| \leq c_6 u \|\mathbf{b}_j\| \leq c_6 u \|\mathbf{b}_j\|_1 = c_6 u \mathbf{e}^T |\mathbf{b}_j|,$$

where $\mathbf{e} = [1, \ldots, 1]^T$. We thus have $|\Delta \mathbf{b}_j| \leq c_6 u \mathbf{e}\mathbf{e}^T |\mathbf{b}_j|$ for all j, which gives $|\Delta B| \leq c_6 u C|B|$ since $C = \mathbf{e}\mathbf{e}^T$. □

6.2. **Application to LLL-reduced matrices.** By using Theorem 6.4 and Corollary 5.7, we have the following result on LLL-reduced bases.

Corollary 6.5. *Let $\eta \in [1/2, 1], \theta \geq 0, \delta \in (\eta^2, 1]$ and $\alpha = \frac{\theta\eta + \sqrt{(1+\theta^2)\delta - \eta^2}}{\delta - \eta^2}$. Let $B \in \mathbb{R}^{m\times n}$ be a (δ, η, θ)-LLL-reduced basis matrix. Let u be such that*

$$u' := c_7 (1 + \eta + \theta)^n \alpha^n u < 1,$$

where $c_7 = c_4 c_6$ and with c_4 defined by (5.13) and c_6 defined by (6.14). Suppose we compute the R-factor of B with the algorithm described in Subsection 6.1. Then the computed matrix $\widehat{R}$ is $(\delta', \eta', \theta')$-LLL-reduced with

$$\delta' = \delta \frac{(1-u')^2}{(1+u')^2(1+2u'(\eta\alpha+\theta))}, \quad \eta' = \frac{\eta}{1-u'}, \quad \theta' = \frac{\theta + u'}{1-u'}.$$

Proof. From Theorem 6.4, we know that (1.1) holds with $\varepsilon = c_6 u$. The result directly follows from Corollary 5.7. □

The weakening of the LLL-reduction is stable under Householder's algorithm: if the input basis is reduced, then so is the output basis (with slightly relaxed factors).

7. Concluding Remarks

We investigated the sensitivity of the R-factor of the QR-factorisation under column-wise perturbations, which correspond to the backward stability results of the standard QR factorization algorithms. We focused on the case of LLL-reduced matrices, and showed that if the classical definition of LLL-reducedness is sligthly modified, then LLL-reducedness is preserved under column-wise perturbations. This implies that by computing the R-factor of a reduced matrix with a

standard floating-point QR factorization algorithm (e.g., Householder's algorithm), then one can numerically check that the LLL conditions (5.3) are indeed satisfied, for slightly degraded parameters. These certified reduction parameters can be made arbitrarily close to the actual reduction parameters by setting the precision sufficiently high. Importantly, the required precision for the above to be valid is linear with respect to the dimension, and does not depend on the magnitudes of the matrix entries. This study was motivated by its algorithmic implications: the results may be used to efficiently check the LLL-reducedness of a basis and to speed up the LLL-reduction process.

References

1. X.-W. Chang and C. C. Paige, *Componentwise perturbation analyses for the QR factorization*, Numerische Mathematik **88** (2001), 319–345.
2. X.-W. Chang, C. C. Paige, and G. W. Stewart, *Perturbation analyses for the QR factorization*, SIAM J. Matrix Anal. Appl. **18** (1997), 775–791.
3. X.-W. Chang and D. Stehlé, *Rigorous perturbation bounds for some matrix factorizations*, SIAM J. Matrix Anal. Appl. **31** (2010), 2841–2859.
4. H. Cohen, *A course in computational algebraic number theory, 2nd edition*, Springer, Berlin, 1995.
5. U. Fincke and M. Pohst, *A procedure for determining algebraic integers of given norm*, Proceedings of EUROCAL, Lecture Notes in Computer Science, vol. 162, 1983, pp. 194–202.
6. G. H. Golub and C. F. Van Loan, *Matrix computations, 3rd edition*, The John Hopkins University Press, Baltimore, MD, 1996.
7. N. J. Higham, *Accuracy and stability of numerical algorithms, 1st edition*, Society for Industrial and Applied Mathematics, 1996.
8. ———, *Accuracy and stability of numerical algorithms, 2nd edition*, Society for Industrial and Applied Mathematics, 2002.
9. R. Kannan, *Improved algorithms for integer programming and related lattice problems*, Proceedings of the 15th Symposium on the Theory of Computing (STOC 1983), ACM Press, 1983, pp. 99–108.
10. D. Knuth, *The analysis of algorithms*, Actes du Congrès International des Mathématiciens de 1970, vol. 3, Gauthiers-Villars, 1971, pp. 269–274.
11. H. Koy and C. P. Schnorr, *Segment LLL-reduction of lattice bases with floating-point orthogonalization*, Proceedings of the 2001 Cryptography and Lattices Conference (CALC'01), Lecture Notes in Computer Science, vol. 2146, Springer, Berlin, 2001, pp. 81–96.
12. A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász, *Factoring polynomials with rational coefficients*, Math. Ann. **261** (1982), 515–534.
13. L. Lovász, *An Algorithmic Theory of Numbers, Graphs and Convexity*, SIAM Publications, 1986, CBMS-NSF Regional Conference Series in Applied Mathematics.
14. I. Morel, D. Stehlé, and G. Villard, *H-LLL: Using Householder inside LLL*, Proceedings of the 2009 International Symposium on Symbolic and Algebraic Computation (ISSAC'09), ACM Press, 2009, pp. 271–278.
15. ———, *From an LLL-reduced basis to another*, Work in progress, 2010.
16. W. H. Mow, *Maximum likelihood sequence estimation from the lattice viewpoint*, IEEE Transactions on Information Theory **40** (1994), 1591–1600.
17. P. Nguyen and D. Stehlé, *Floating-point LLL revisited*, Proceedings of Eurocrypt 2005, Lecture Notes in Computer Science, vol. 3494, Springer, Berlin, 2005, pp. 215–233.
18. ———, *An LLL algorithm with quadratic complexity*, SIAM Journal on Computing **39** (2009), no. 3, 874–903.
19. P. Nguyen and J. Stern, *The two faces of lattices in cryptology*, Proceedings of the 2001 Cryptography and Lattices Conference (CALC'01), Lecture Notes in Computer Science, vol. 2146, Springer, Berlin, 2001, pp. 146–180.
20. P. Q. Nguyen and B. Vallée (editors), *The LLL algorithm: survey and applications*, Information Security and Cryptography, Springer, Berlin, 2009, Published after the LLL25 conference held in Caen in June 2007, in honour of the 25-th anniversary of the LLL algorithm.

21. A. Novocin, D. Stehlé, and G. Villard, *An LLL-reduction algorithm with quasi-linear time complexity*, Accepted to STOC 2011.
22. C. P. Schnorr, *A more efficient algorithm for lattice basis reduction*, Journal of Algorithms **9** (1988), no. 1, 47–62.
23. ———, *Fast LLL-type lattice reduction*, Inform. Comput. **204** (2006), 1–25.
24. ———, *Progress on LLL and lattice reduction*, In [20], 2009.
25. A. Schönhage, *Schnelle Berechnung von Kettenbruchentwicklungen*, Acta Informatica **1** (1971), 139–144.
26. A. Schönhage and V. Strassen, *Schnelle Multiplikation grosser Zahlen*, Computing **7** (1971), 281–292.
27. G. Villard, *Certification of the QR factor R and of lattice basis reducedness*, Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation (ISSAC'07), ACM Press, 2007, pp. 143–150.
28. H. Zha, *A componentwise perturbation analysis of the QR decomposition*, SIAM J. Matrix Anal. Appl. **14** (1993), no. 4, 1124–1131.

School of Computer Science, McGill University Montreal, Quebec, Canada H3A 2A7, http://www.cs.mcgill.ca/~chang
E-mail address: chang@cs.mcgill.ca

CNRS, Université de Lyon, Laboratoire LIP, CNRS-ENSL-INRIA-UCBL, 46 Allée d'Italie, 69364 Lyon Cedex 07, France, http://perso.ens-lyon.fr/damien.stehle
E-mail address: damien.stehle@ens-lyon.fr

CNRS, Université de Lyon, Laboratoire LIP, CNRS-ENSL-INRIA-UCBL, 46 Allée d'Italie, 69364 Lyon Cedex 07, France, http://perso.ens-lyon.fr/gilles.villard
E-mail address: gilles.villard@ens-lyon.fr

H-LLL: Using Householder Inside LLL

Ivan Morel
ÉNS Lyon, Université de Lyon
University of Sydney
Laboratoire LIP, France
CNRS-ENSL-INRIA-UCBL
ivan.morel@ens-lyon.fr

Damien Stehlé
CNRS, Macquarie University
and University of Sydney
Department of Mathematics
and Statistics
University of Sydney
NSW 2006, Autralia
damien.stehle@gmail.com

Gilles Villard
CNRS, Université de Lyon
Laboratoire LIP, France
CNRS-ENSL-INRIA-UCBL
gilles.villard@ens-lyon.fr

ABSTRACT

We describe a new LLL-type algorithm, H-LLL, that relies on Householder transformations to approximate the underlying Gram-Schmidt orthogonalizations. The latter computations are performed with floating-point arithmetic. We prove that a precision essentially equal to the dimension suffices to ensure that the output basis is reduced. H-LLL resembles the L^2 algorithm of Nguyen and Stehlé that relies on a floating-point Cholesky algorithm. However, replacing Cholesky's algorithm by Householder's is not benign, as their numerical behaviors differ significantly. Broadly speaking, our correctness proof is more involved, whereas our complexity analysis is more direct. Thanks to the new orthogonalization strategy, H-LLL is the first LLL-type algorithm that admits a natural vectorial description, which leads to a complexity upper bound that is proportional to the progress performed on the basis (for fixed dimensions).

Categories and Subject Descriptors

F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems—*Computations on matrices*

General Terms

Algorithms

Keywords

Lattice Reduction, LLL, Floating-Point Arithmetic, Householder's Algorithm

1. INTRODUCTION

Lattice reduction is a fundamental tool in diverse fields of computational mathematics [2] and computer science [8]. The LLL algorithm, invented in 1982 by Arjen Lenstra, Hendrik Lenstra Jr and László Lovász [7], allows one to perform lattice reduction in time polynomial in both the dimensions and the bit-sizes of the entries of the input matrix.

In terms of efficiency, the major weakness of the original rational algorithm and its improved variants [5, 17] is that they perform all computations with exact arithmetic, leading to the use of very large integers. This considerably slows down the algorithm, making it impractical for large dimensions or entries. As early as 1983, Odlyzko, in his first attempts to cryptanalyze knapsack cryptosystems [10], used floating-point arithmetic (fpa for short) within LLL to avoid the rational arithmetic cost overhead. The cost of updating the basis being negligible compared to the cost of computing and updating the Gram-Schmidt orthogonalization (GSO for short) of the vectors, it seems natural to compute the latter using fpa, while using exact arithmetic to update the basis. This was at first implemented in a heuristic manner, without ensuring the accuracy of the computations. In a pioneering work [13], Schnorr showed that the natural heuristic approach can be made rigorous.

In the present paper we present a new fp LLL algorithm that relies on the computation of the QR-factorization of the basis using Householder's algorithm. H-LLL computes fp approximations to the coefficients of the R-factor and uses them to perform exact operations on the basis. We prove that if the precision is large enough, then H-LLL runs correctly. The bound on the precision depends on the dimension only (it is actually essentially equal to it). Our analysis relies on bounds on the errors made while computing the R-factor of a given reduced basis. Those bounds are proved in [1]. Exploiting them while requiring a fairly small precision is where the technical complexity of the present work lies. In particular, the bounds do not seem sufficient to perform a size-reduction, a crucial step in the LLL algorithm (even with the weaker version of Definition 2). This is where H-LLL differs from most LLL variants: rather than fully size-reducing the current vector, we transform it so that enough information is obtained to decide whether Lovász's condition is satisfied. The correctness of H-LLL is thus harder to prove, but its unique design allows us to explicitly bound the bit-complexity in terms of the actual work that was performed on the lattice basis. All other LLL algorithms work on the underlying quadratic form, whereas ours can be interpreted as working on vectors. Considering a basis matrix $(\mathbf{b}_1, \ldots, \mathbf{b}_d) \in \mathbb{Z}^{n \times d}$ with vectors of euclidean norms $\leq \|B\|$, the total bit complexity is:

$$O\left[\left(d + \log \prod \frac{d_i^b}{d_i^e} + \frac{1}{d} \log \prod \frac{\|\mathbf{b}_i^b\|}{\|\mathbf{b}_i^e\|}\right) n\mathcal{M}(d)(d + \log \|B\|)\right],$$

ISSAC'09, July 28–31, 2009, Seoul, Republic of Korea.

where d_i^b (resp. d_i^e) is the determinant of the lattice spanned by the first i columns of B at the beginning (resp. the end), and $\mathcal{M}(x) = O(x^2)$ is the cost of multiplying two x-bit long integers. The product $\prod d_i$ is classically referred to as the potential. The term $\log \prod \frac{d_i^b}{d_i^e}$ quantifies the actual progress made with respect to the potential, while the term $\log \prod \frac{\|\mathbf{b}_i^b\|}{\|\mathbf{b}_i^e\|}$ quantifies the progress made with respect to the norms of the vectors. One can note that the obvious bound on the latter ($d \log \|B\|$) is negligible compared to the obvious bound on the former ($d^2 \log \|B\|$). The overall bit complexity is $O(nd^2\mathcal{M}(d) \log \|B\|(d + \log \|B\|))$.

Related works. As mentioned previously, the first rigorous fp LLL was invented by Schnorr in 1988 (see [13]). However, the precision used in the fp computations was a linear function of both the bit-size of the matrix entries and the dimension, with rather large constant factors. Since then, Schnorr *et. al* have described several heuristic reduction algorithms [15, 6, 14, 12], notably introducing in [15] the concept of lazy size-reduction and in [6] the idea to use Householder's algorithm. The outputs of those heuristic algorithms may be certified LLL-reduced with [18], but so far there does not exist any proved variant of LLL relying on Householder's algorithm and using a fp precision that does not depend on the bit-size of the matrix entries. The L^2 algorithm [9] of Nguyen and Stehlé is a proven fp LLL, also of complexity $O(nd^2\mathcal{M}(d) \log \|B\|(d + \log \|B\|))$, that relies on a lazy size-reduction based on Cholesky's algorithm. Although this approach is close to the present work, there are a few key differences caused by the use of different orthogonalization algorithms. The first difference is the nature of the numerical errors. Both Cholesky's algorithm and Householder's are backward stable [4] and forward stable when the input is LLL-reduced [9, 1]. When computing the R-factor of a given basis, the error made using Cholesky's relates to the diagonal coefficient of the row, which induces an absolute error on the Gram-Schmidt coefficients. When using Householder's, the same error involves the diagonal coefficient of the column, inducing possibly much larger absolute errors on the Gram-Schmidt coefficients. This leads us to use a slightly relaxed definition of reduction, which is a fixpoint under perturbation of the original basis [1]. The different nature of the error makes the correctness harder to obtain. The second difference is the number and type of arithmetic operations made. Cholesky's algorithm uses the exact Gram matrix of the basis to compute the R-factor, which implies additional integer arithmetic. Furthermore the overall number of operations needed to compute and update the GSO-related quantities using Cholesky's algorithm is roughly twice the number of operations needed when using Householder's. Also, the precision required is higher when using the Cholesky factorization, which can be explained intuitively by its condition number being greater than the condition number of the QR-factorization. This leads to the fact that H-LLL requires a precision of $\approx d$ bits, whereas L^2 requires a precision of $\approx 1.6d$ bits. Finally, the vectorial nature of H-LLL makes its complexity analysis simpler than that of L^2: the amortized cost analysis (which allows to get a complexity bound that is quadratic when the dimensions are fixed) is much more direct.

Road-map. In Section 2, we give some reminders that are necessary for the description and analysis of H-LLL. In Section 3, we describe a new (incomplete) size-reduction algorithm and analyze it. H-LLL relies on the (incomplete) size-reduction algorithm and is presented in Section 4.

Notation. Vectors will be denoted in bold. If $\mathbf{b}$ is a vector, then $\|\mathbf{b}\|$ will denote its euclidean norm. For a matrix $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$, its j-th column will be denoted by $\mathbf{a}_j$. If $\mathbf{b}$ is a vector and $i \leq j$ are two valid entry indices, then $\mathbf{b}[i..j]$ is the $(j-i+1)$-dimensional sub-vector of $\mathbf{b}$ consisting of its entries within indices i and j. The notation $\lfloor x \rceil$ denotes an arbitrary integer closest to x. We define $\mathrm{sign}(x)$ as 1 if $x \geq 0$ and -1 otherwise. We use a standard base-2 arbitrary precision fp model, such as described in [4, Sec. 2.1]. The notation $\diamond(a)$ refers to the fp rounding of a. If x is a variable, the variable $\overline{x}$ hopefully approximates x and Δx is the distance between them. For complexity statements, we count all elementary bit operations.

Glossary. The variables $\alpha, \delta, \overline{\delta}, \delta', \eta, \overline{\eta}, \theta, \overline{\theta}$ and ρ all refer to parameters related to the LLL-reduction. For simplicity, the reader may think of $\alpha \approx 2/\sqrt{3}$, $1 \approx \delta < \overline{\delta} < \delta' < 1$, $1/2 < \overline{\eta} < \eta \approx 1/2$, $0 < \overline{\theta} < \theta \approx 0$ and $\rho \approx \sqrt{3}$. The variables c_0, c_1 are polynomially bounded functions of d and n (and the variables above) and can be safely thought of as constants.

2. LATTICE REDUCTION

A euclidean lattice L is a discrete subgroup of $\mathbb{R}^n$. A basis $B = (\mathbf{b}_1, \ldots, \mathbf{b}_d) \in L^d$ of L is a tuple of linearly independent vectors such that L is precisely the set of all integer linear combinations of the $\mathbf{b}_i$'s. The integer $d \leq n$ is the dimension of L. Any lattice L of dimension $d \geq 2$ has infinitely many bases, which can all be derived from any arbitrary basis of L by applying unimodular transformations, i.e., invertible integral operations. Lattice reduction aims at finding 'good' bases, i.e., bases with reasonably short and orthogonal vectors. Having such a basis allows one to obtain information about the lattice more easily. In the following we consider only integer lattices, i.e., $L \subseteq \mathbb{Z}^n$. We represent a basis B by using the $n \times d$ integer matrix whose columns are the $\mathbf{b}_i$'s. We will now introduce some elementary notions about lattices. We refer to [8] for more details.

Orthogonalization. The Gram-Schmidt orthogonalization maps a basis $B = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$ to a tuple of orthogonal vectors $(\mathbf{b}_1^*, \ldots, \mathbf{b}_d^*)$ defined by:

$$\forall i \leq d,\ \mathbf{b}_i^* = \mathbf{b}_i - \sum_{j<i} \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\|\mathbf{b}_j^*\|^2} \mathbf{b}_j^*.$$

The GSO quantifies the orthogonality of the $\mathbf{b}_i$'s. If the $\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle / \|\mathbf{b}_j^*\|^2$'s are small and the $\|\mathbf{b}_i^*\|$'s do not decrease too fast, then the $\mathbf{b}_i$'s are fairly orthogonal. The GSO is closely related to the R-factor of the QR-factorization of the basis matrix. For a given $B \in \mathbb{R}^{n \times d}$ of rank d, there exist matrices $Q \in \mathbb{R}^{n \times d}$ and $R \in \mathbb{R}^{d \times d}$, such that $Q^T Q = I$, R is upper triangular with positive diagonal coefficients and $B = QR$. Such a factorization is unique and we have $R_{i,i} = \|\mathbf{b}_i^*\|$ and $R_{i,j} = \langle \mathbf{b}_j, \mathbf{b}_i^* \rangle / \|\mathbf{b}_i^*\|$ for any $i < j$.

Lattice invariants. An invariant of a lattice L is a quantity that does not depend on the particular choice of a basis of L. The minimum is defined by: $\lambda_L = \min(\|\mathbf{b}\|, \mathbf{b} \in L \setminus \{\mathbf{0}\})$. The determinant $\det L = \sqrt{\det(B^T B)} = \prod \|\mathbf{b}_i^*\|$ is another lattice invariant.

LLL-reduction. The LLL-reduction is an efficiently computable relaxation of a reduction introduced by Hermite [3]. We give a generalization of the definition of [7].

Definition 1. Let $\eta \geq 1/2$ and $\delta \leq 1$. A basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ is (δ, η)-LLL reduced if for any $i < j$, $|R_{i,j}| \leq \eta R_{i,i}$ (size-reduction condition) and if for any i, $\delta R_{i-1,i-1}^2 \leq R_{i-1,i}^2 + R_{i,i}^2$ (Lovász's condition).

For the purpose of this work, we need a slightly weaker definition of reduction, introduced in [1]. One can recover Definition 1 by taking $\theta = 0$.

Definition 2. Let $\eta \geq 1/2$, $\delta \leq 1$ and $\theta \geq 0$. A basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ is (δ, η, θ)-LLL reduced if for any $i < j$, $|R_{i,j}| \leq \eta R_{i,i} + \theta R_{j,j}$ (weak size-reduction condition) and if Lovász's condition holds.

The latter definition is essentially equivalent to the former, as it only differs when $R_{j,j} \gg R_{i,i}$, which corresponds to quite orthogonal vectors. The following theorem (from [1]) formalizes this equivalence by exhibiting properties of (δ, η, θ)-reduced bases similar to the properties of (δ, η)-reduced bases [7].

THEOREM 2.1. *Let $\eta \in [1/2, 1]$, $\theta \geq 0$, $\delta \in (\eta^2, 1]$ and $\alpha = \frac{\theta\eta + \sqrt{(1+\theta^2)\delta - \eta^2}}{\delta - \eta^2}$. Let $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ be a (δ, η, θ)-LLL reduced basis of a lattice L. Then for all i, we have $R_{i,i} \leq \alpha R_{i+1,i+1}$ and $R_{i,i} \leq \|\mathbf{b}_i\| \leq \alpha^{i-1} R_{i,i}$. We also have $\|\mathbf{b}_1\| \leq \alpha^{d-1}\lambda_L$, $\|\mathbf{b}_1\| \leq \alpha^{\frac{d-1}{2}} (\det L)^{\frac{1}{d}}$ and $\prod \|\mathbf{b}_i\| \leq \alpha^{\frac{d(d-1)}{2}} (\det L)$.*

The LLL algorithm. LLL [7] computes a (δ, η)-LLL-reduced basis in time polynomial both in the dimensions d and n and the bit-size of the entries $\log \|B\|$, provided that $\eta \in [1/2, 1)$ and $\delta \in (\delta - \eta^2, 1)$. Although there are many LLL variants, they all roughly follow the same high-level design, described in Algorithm 1.

Algorithm 1 A generic LLL algorithm.

Input: A basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ of a lattice L.
Output: An LLL-reduced basis of L.

```
κ := 2.
While κ ≤ d, do
    Size-reduce b_κ.
    If Lovász's condition holds for κ, then κ := κ + 1.
    Else swap b_{κ−1} and b_κ; κ := max(κ − 1, 2).
```

Perturbation analysis of the R-factor. In this paper we introduce a new variant of LLL that relies on the approximate computation of the R-factor of B using Householder's algorithm (Algorithm 2). With fpa, all operations are performed in the naive order, and all sums of several terms are computed sequentially. In order to ensure the soundness of the operations we will perform on the basis (in H-LLL), which are dictated by the values of the $R_{i,j}$, we need to address the issue of the accuracy of the computed R-factor. It is known (see [4, Ch. 19]) that Householder's algorithm computing the R-factor is backward-stable (i.e., its output is the R-factor of a matrix that is close to its input), but it is not forward-stable in the general case. Theorem 2.3 (proved in [1]) bounds the sensibility of the R-factor to column-wise input perturbations, when the input is LLL-reduced. Combined with the backward stability of Householder's algorithm (Theorem 2.2, proved in [1]), Corollary 2.4 shows the forward-stability of Householder's algorithm in the case of LLL-reduced inputs.

Algorithm 2 Householder's algorithm.

Input: A rank d matrix $B \in \mathbb{R}^{n \times d}$.
Output: An approximation to the R-factor of B.

```
R := ⋄(B).
For i from 1 to d, do
    For j from 1 to i − 1, do
        r_i[j..n] = r_i[j..n] − (v_j^T r_i[j..n]) · v_j; r_i[j] := σ_j r_i[j].
    r := r_i[i..n]; v_i := r.
    σ_i := sign(r[1]); s := σ_i ‖r‖.
    v_i[1] := (− Σ_{j=2}^{n−i+1} r[j]^2)/(r[1] + s).
    If v_i[1] ≠ 0, then v_i := v_i/√(−s · v_i[1]).
    r_i[i..n] := (‖r‖, 0, ..., 0)^T.
Return the first d rows of R.
```

THEOREM 2.2. *Let $B \in \mathbb{R}^{n \times d}$ be a rank d matrix given as input to Algorithm 2. Let us assume that the computations are performed with fpa in precision p such that $8d(n+9)2^{-p} \leq 1$. Let $\overline{R} \in \mathbb{R}^{d \times d}$ be the output. Then there exists $Q \in \mathbb{R}^{n \times d}$ with orthonormal columns such that $\Delta B = B - Q\overline{R}$ satisfies:*

$$\forall i \leq d,\ \Delta\|\mathbf{b}_i\| \leq 8d(n+9)2^{-p} \cdot \|\mathbf{b}_i\|.$$

THEOREM 2.3. *Let $\eta \in [1/2, 1), \theta \geq 0$ and $\delta \in (\eta^2, 1]$. Let $B \in \mathbb{R}^{n \times d}$ of rank d be (δ, η, θ)-LLL-reduced. Let $\varepsilon \geq 0$ such that $c_0 \rho^d \varepsilon < 1$, where $\rho = (1 + \eta + \theta)\alpha$ and:*

$$c_0 = \max\left\{ \frac{1 + |1 - \eta - \theta|\alpha}{(\eta + \theta)\left(-1 + \sqrt{\frac{3}{2}}\right)}, \frac{4\sqrt{6}}{1+\eta}\sqrt{1 + d\eta^2} \right\} n\sqrt{d}.$$

If $\Delta B \in \mathbb{R}^{n \times d}$ is such that $\forall i, \Delta\|\mathbf{b}_i\| \leq \varepsilon \cdot \|\mathbf{b}_i\|$ and if $R + \Delta R$ is the R-factor of $B + \Delta B$ (which exists), then:

$$\forall i \leq d,\ \Delta\|\mathbf{r}_i\| \leq c_0 \rho^i \varepsilon \cdot R_{i,i}.$$

The following result provides an error bound for the $\overline{R}$ matrix computed by Algorithm 2 using precision p fpa, starting from a B in $\mathbb{R}^{n \times d}$ whose $d-1$ first columns are LLL-reduced.

COROLLARY 2.4. *Let $\eta \in [1/2, 1), \theta \geq 0$ and $\delta \in (\eta^2, 1)$. Let $B \in \mathbb{R}^{n \times d}$ be a rank d matrix whose first $(d-1)$ columns are (δ, η, θ)-LLL-reduced and which is given as input to Algorithm 2. Let us assume that the computations are performed with fpa in precision p such that $c_1 \rho^d 2^{-p} < 1$, where $c_1 = 8d(n+9)c_0$. Let $\overline{R} = R + \Delta R \in \mathbb{R}^{d \times d}$ be the output matrix. Then:*

$$\forall j \leq i < d,\ \Delta R_{j,i} \leq c_1 \rho^i 2^{-p} \cdot R_{i,i}$$

and

$$\forall i < d,\ \Delta R_{i,d} \leq c_1(1 + 1/\theta)\rho^{i+1} 2^{-p} \cdot (R_{i,i} + \|\mathbf{b}_d\|).$$

Thus denoting the quantity $c_1(1+1/\theta)\rho^{i+1}$ by $\phi(i)$, we have for any $j \leq i < d$:

$$\Delta R_{j,i} \leq 2^{-p}\phi(i) R_{i,i} \text{ and } \Delta R_{i,d} \leq 2^{-p}\phi(i)(R_{i,i} + \|\mathbf{b}_d\|).$$

Proof. The first statement is a direct consequence of Theorems 2.2 and 2.3. Let $i < d$. We consider the basis $(\mathbf{b}'_1, \ldots, \mathbf{b}'_{i+1})$ defined by $\mathbf{b}'_j = (\mathbf{b}_j^T, 0)^T$ for $j \leq i$ and $\mathbf{b}'_{i+1} = (\mathbf{b}_d^T, R_{i,i} + \|\mathbf{b}_d\|/\theta)^T$. By construction, it is (δ, η, θ)-LLL reduced. Furthermore, calling Algorithm 2 on $(\mathbf{b}'_1, \ldots, \mathbf{b}'_{i+1})$ leads to exactly the same fp operations as on $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$, for the approximation of $R'_{i,i+1} = R_{i,d}$. Therefore, using the first part of the result:

$$\Delta R_{i,d} = \Delta R'_{i,i+1} \leq c_1 \rho^{i+1} 2^{-p} \cdot R'_{i+1,i+1}.$$

Then we use $R'_{i+1,i+1} \leq R_{i,i} + (1 + 1/\theta)\|\mathbf{b}_d\|$. □

This result implies that if we start from a (δ, η, θ)-LLL-reduced basis, then we can use Householder's algorithm to check that it is reduced for (arbitrarily) slightly weaker parameters. It is incorrect to say that if we start from a (δ, η)-reduced basis, then Householder's algorithm allows to check that it is (δ', η')-reduced for slightly weaker parameters δ' and η' (a counter-example is provided in [16]). This is the reason that underlies the weakening of the LLL-reduction.

3. AN INCOMPLETE SIZE-REDUCTION

In the present section, we present a novel algorithm (Algorithm 3) that relies on a fp Householder's algorithm (Algorithm 2). It does not size-reduce the vector $\mathbf{b}_\kappa$ under scope, it does not even weakly size-reduce it in general. However, to some extent, it decreases the length of $\mathbf{b}_\kappa$. This is exactly the progress it attempts to make (see Step 7). Also, we will prove that the output basis is of sufficient numerical quality for Lovász's condition to be (approximately) tested. If the latter is satisfied, then we know a posteriori that the basis was indeed weakly size-reduced (see Section 4). The condition on the precision p ensures the soundness of the computations.

The algorithm contains a main loop (Steps 1–7). The vector $\mathbf{b}_\kappa$ becomes more reduced with respect to the previous ones every time the loop is iterated. Within the loop, Householder's algorithm is called (Step 2) to obtain an approximation to $\mathbf{r}_\kappa$. This approximation is then used to perform a partial size-reduction (Steps 3–6), whose progress may be limited by the inaccuracies created at Step 2. Note that only the GSO computations are performed approximately, the basis operations being always exact. Right before the end, at Step 8, new approximations $\overline{\mathbf{r}}_\kappa$ and $\overline{\mathbf{v}}_\kappa$ are computed to ensure that the output vectors $\overline{\mathbf{r}}_1, \ldots, \overline{\mathbf{r}}_\kappa$ and $\overline{\mathbf{v}}_1, \ldots, \overline{\mathbf{v}}_\kappa$ are exactly those that would have been returned by Algorithm 2 given the first κ columns of the returned B as input.

During the execution, the quantities $R_{i,\kappa}$ for $i < \kappa$ are known only approximately, and are updated within the loop made of Steps 3–5. To simplify the exposure, we introduce some notation. We will denote by $\overline{R}_{i,\kappa}$ (resp. $R_{i,\kappa}$) the approximate (resp. exact) value of $R_{i,\kappa}$ at Step 2. We will denote by $\overline{R}'_{i,\kappa}$ the approximate value of $R_{i,\kappa}$ at the beginning of Step 4. This is an approximation to $R'_{i,\kappa} = R_{i,\kappa} - \sum_{j=i+1}^{\kappa-1} X_j R_{i,j}$. Finally, we define $R''_{i,\kappa} = R'_{i,\kappa} - X_i R_{i,i}$, which is the new (exact) value of $R_{i,\kappa}$ after Step 4. We will also use the index i_0 to denote the largest $i < \kappa$ such that $X_i \neq 0$, with $i_0 = 0$ if not defined.

We analyze Algorithm 3 as follows. We first consider the effect of one iteration of the loop made of Steps 3–6 on the $R_{i,\kappa}$'s and $\|\mathbf{b}_\kappa\|$. This study will then lead us to correctness and complexity results on Algorithm 3.

Algorithm 3 The incomplete size-reduction algorithm.

Input: A matrix $B \in \mathbb{Z}^{n \times d}$, $\kappa \leq d$ and the output $\overline{\mathbf{r}}_1, \ldots, \overline{\mathbf{r}}_{\kappa-1}, \overline{\mathbf{v}}_1, \ldots, \overline{\mathbf{v}}_{\kappa-1}, \sigma_1, \ldots, \sigma_{\kappa-1}$ of Algorithm 2 when given as input the first $\kappa - 1$ columns of B. We assume that the first $\kappa - 1$ columns of B are (δ, η, θ)-LLL-reduced with $\eta \in (1/2, 1)$, $\delta \in (\eta^2, 1)$ and $\theta \in (0, \eta - 1/2)$.

Input: $\diamond(2^{-cd})$ (for an arbitrary $c > 0$) and a fp precision $p > \log_2(2^{\frac{cd}{2}+9}\kappa^3\phi(\kappa)\alpha/\theta)$.

1: Do
2: Compute $\overline{\mathbf{r}}_\kappa$ using Steps 3–4 of Algorithm 2.
3: For i from $\kappa - 1$ to 1, do
4: $X_i = \left\lfloor \diamond\left(\frac{\overline{R}_{i,\kappa}}{\overline{R}_{i,i}}\right) \right\rceil$.
5: For j from 1 to $i-1$, do $\overline{R}_{j,\kappa} := \diamond\left(\overline{R}_{j,\kappa} - \diamond\left(X_i \overline{R}_{j,i}\right)\right)$.
6: $t := \diamond(\|\mathbf{b}_\kappa\|^2)$; $\mathbf{b}_\kappa := \mathbf{b}_\kappa - \sum_{i<\kappa} X_i \mathbf{b}_i$.
7: Until $\diamond(\|\mathbf{b}_\kappa\|^2) > \diamond(\diamond(2^{-cd}) \cdot t)$.
8: Compute $\overline{\mathbf{r}}_\kappa, \overline{\mathbf{v}}_\kappa, \sigma_\kappa$ using Steps 3–9 of Algorithm 2.
9: Return B, $\overline{\mathbf{r}}_1, \ldots, \overline{\mathbf{r}}_\kappa$, $\overline{\mathbf{v}}_1, \ldots, \overline{\mathbf{v}}_\kappa$ and $\sigma_1, \ldots, \sigma_\kappa$.

3.1 Analysis of Steps 3–6

The aim of the next lemmata is to bound the magnitude of $R'_{i,\kappa}$ and its error $\Delta R'_{i,\kappa}$. As is often the case in numerical analysis, the error and magnitude bounds are intertwined. This issue is solved by building up an induction on the two bounds (Lemmata 3.2 and 3.3), and the induction itself is solved in Lemma 3.4. This allows us to lower bound the decrease of $\|\mathbf{b}_\kappa\|$ after an iteration of the loop (in Theorem 3.7).

LEMMA 3.1. *For any $i < \kappa$, the quantity $|X_i| R_{i,i}$ is upper bounded by both*

$$\frac{R_{i,i}}{2} + (1 + 2^{-p+1}\phi(i))|\overline{R}'_{i,\kappa}| \quad \text{and} \quad 4|\overline{R}'_{i,\kappa}|.$$

Proof. The result being obviously correct when $X_i = 0$, we assume that $X_i \neq 0$. We have that $|X_i|$ is no greater than

$$1/2 + \diamond(|\overline{R}'_{i,\kappa}|/\overline{R}_{i,i}) \leq 1/2 + (1 + 2^{-p})|\overline{R}'_{i,\kappa}|/\overline{R}_{i,i}.$$

Therefore, by using Corollary 2.4:

$$\begin{aligned} |X_i||R_{i,i}| &\leq \frac{R_{i,i}}{2} + \frac{1 + 2^{-p}}{1 - 2^{-p}\phi(i)}|\overline{R}'_{i,\kappa}| \\ &\leq \frac{R_{i,i}}{2} + (1 + 2^{-p+1}\phi(i))|\overline{R}'_{i,\kappa}|. \end{aligned}$$

Since $X_i \neq 0$, we have $|\overline{R}'_{i,\kappa}| \geq \frac{\overline{R}_{i,i}}{2} \geq \frac{(1-2^{-p}\phi(i))R_{i,i}}{2}$. Thus:

$$|X_i||R_{i,i}| \leq 2(1 + 2^{-p+1}\phi(i))|\overline{R}'_{i,\kappa}|,$$

which completes the proof. □

LEMMA 3.2. *For any $i \leq i_0$, we have:*

$$\begin{aligned} |R'_{i,\kappa}| \leq &\|\mathbf{b}_\kappa\| + \kappa\alpha^{i_0-i} R_{i_0,i_0} \\ &+ (1 + 2^{-p+1}\phi(i_0)) \sum_{j=i+1}^{i_0} \left(\eta\alpha^{j-i} + \theta\right) |\overline{R}'_{j,\kappa}|. \end{aligned}$$

Proof. By using the LLL-reducedness of the first $\kappa - 1$

columns of B, we have:

$$\begin{aligned}|R'_{i,\kappa}| &\leq |R_{i,\kappa}| + \sum_{j=i+1}^{i_0} |X_j||R_{i,j}| \\ &\leq \|\mathbf{b}_\kappa\| + \sum_{j=i+1}^{i_0} (\eta\alpha^{j-i} + \theta)|X_j|R_{j,j} \\ &\leq \|\mathbf{b}_\kappa\| + \kappa\alpha^{i_0-i}R_{i_0,i_0}.\end{aligned}$$

The result is then provided by Lemma 3.1. □

LEMMA 3.3. *For any $i \leq i_0$, we have:*

$$\Delta R'_{i,\kappa} \leq 2^{-p+2}\phi(i)(\|\mathbf{b}_\kappa\| + R_{i,i}) + 2^{-p+4}\sum_{j=i+1}^{i_0} \phi(j)|\overline{R}'_{j,\kappa}|.$$

Proof. Using the bound [4, Eq. (3.5)], Corollary 2.4, Lemma 3.1 and the LLL-reducedness of the first $\kappa - 1$ columns of B, we have that $\Delta R'_{i,\kappa}$ is bounded by:

$$\begin{aligned}&\kappa 2^{-p+1}\left(|\overline{R}_{i,\kappa}| + \sum_{j=i+1}^{i_0} |X_j\overline{R}_{i,j}|\right) + \sum_{j=i+1}^{i_0} |X_j|\Delta R_{i,j} + \Delta R_{i,\kappa} \\ &\leq \kappa 2^{-p+1}\left(\|\mathbf{b}_\kappa\| + \sum_{j=i+1}^{i_0} |X_j R_{i,j}|\right) + 2\sum_{j=i+1}^{i_0} |X_j|\Delta R_{i,j} + 2\Delta R_{i,\kappa} \\ &\leq \kappa 2^{-p+1}\|\mathbf{b}_\kappa\| + 2^{-p+1}\sum_{j=i+1}^{i_0} |X_j|(\kappa R_{i,i} + \phi(j)R_{j,j}) + 2\Delta R_{i,\kappa} \\ &\leq \kappa 2^{-p+1}\|\mathbf{b}_\kappa\| + 2^{-p+1}\phi(i)(\|\mathbf{b}_\kappa\| + R_{i,i}) \\ &\quad + 2^{-p+3}\sum_{j=i+1}^{i_0} (\kappa\alpha^{j-i} + \phi(j))|\overline{R}'_{j,\kappa}|,\end{aligned}$$

which provides the result. □

LEMMA 3.4. *For any $i \leq i_0$, we have that $|\overline{R}'_{i,\kappa}| \leq 2\kappa\rho^{i_0-i}(\|\mathbf{b}_\kappa\| + R_{i_0,i_0})$. This bound also holds for any $|\overline{R}_{i,\kappa}|$ at any moment within the loop made of Steps 3–5.*

Proof. Using Lemmata 3.2 and 3.3, we bound $|\overline{R}'_{i,\kappa}|$ by:

$$\begin{aligned}&|R'_{i,\kappa}| + \Delta R'_{i,\kappa} \\ &\leq |R'_{i,\kappa}| + 2^{-p+2}\phi(i)(\|\mathbf{b}_\kappa\| + R_{i,i}) + 2^{-p+4}\sum_{j=i+1}^{i_0} \phi(j)|\overline{R}'_{j,\kappa}| \\ &\leq \alpha\|\mathbf{b}_\kappa\| + 2\kappa\alpha^{i_0-i}R_{i_0,i_0} \\ &\quad + \sum_{j=i+1}^{i_0} \left(\eta\alpha^{j-i} + \theta + 2^{-p+5}\phi(i_0)\alpha^{j-i}\right)|\overline{R}'_{j,\kappa}|.\end{aligned}$$

We now define $(u_i)_{i\leq i_0}$ by $u_{i_0} = |\overline{R}_{i_0,\kappa}|$ and, for $i < i_0$:

$$u_i = \alpha\|\mathbf{b}_\kappa\| + 2\kappa\alpha^{i_0-i}R_{i_0,i_0} + \sum_{j=i+1}^{i_0} A(i,j)u_j,$$

with $A(i,j) = \eta\alpha^{j-i} + \theta + 2^{-p+5}\phi(i_0)\alpha^{j-i}$. For any $i \leq i_0$, we have $|\overline{R}'_{i,\kappa}| \leq u_i$. Moreover, using the fact that $R_{i,i} \leq \alpha R_{i+1,i+1}$, we obtain that for $i < i_0 - 1$:

$$u_i - \alpha u_{i+1} \leq A(i,i+1)u_{i+1} \leq \alpha(\eta+\theta)u_{i+1}.$$

Thus $u_i \leq \rho u_{i+1}$ and, by using Corollary 2.4, we have that for any $i < i_0$:

$$\begin{aligned}u_i &\leq \rho^{i_0-i-1}u_{i_0-1} \\ &\leq \rho^{i_0-i-1}\alpha\left(\|\mathbf{b}_\kappa\| + 2\kappa R_{i_0,i_0} + (\eta+\theta)\left(\|\mathbf{b}_\kappa\| + \Delta R_{i_0,\kappa}\right)\right) \\ &\leq 2\rho^{i_0-i-1}\left(\rho\|\mathbf{b}_\kappa\| + \kappa\alpha R_{i_0,i_0} + \alpha(\eta+\theta)2^{-p}\phi(i_0)R_{i_0,i_0}\right),\end{aligned}$$

which gives the result for $i < i_0$. To conclude, note that:

$$u_{i_0} \leq \|\mathbf{b}_\kappa\| + \Delta R_{i_0,\kappa} \leq 2(\|\mathbf{b}_\kappa\| + 2^{-p}\phi(i_0)R_{i_0,i_0}).$$

This completes the proof. □

We can now use Lemma 3.4 to obtain a bound on the $\Delta R'_{i,\kappa}$'s that does not depend on the computed $\overline{R}'_{i,\kappa}$'s but only on their exact values.

LEMMA 3.5. *For any $i \leq i_0$, we have:*

$$\Delta R'_{i,\kappa} \leq 2^{-p+6}\kappa^2\phi(i_0)(\|\mathbf{b}_\kappa\| + R_{i_0,i_0}).$$

Proof. Using Lemma 3.4, we have:

$$\begin{aligned}\sum_{j=i+1}^{i_0} \phi(j)|\overline{R}'_{j,\kappa}| &\leq 2\kappa(\|\mathbf{b}_\kappa\| + R_{i_0,i_0})\sum_{j=i+1}^{i_0-1} \phi(j)\rho^{i_0-j} \\ &\leq 2\kappa^2(\|\mathbf{b}_\kappa\| + R_{i_0,i_0})\phi(i_0).\end{aligned}$$

Together with Lemma 3.3, the latter provides the result. □

Now that we understand precisely the $R'_{i,\kappa}$'s, we study the $R''_{i,\kappa}$'s.

LEMMA 3.6. *Let $\bar{\eta} = 1/2 + 2^{-p+1}\phi(\kappa)$. We have:*

$$|R''_{i,\kappa}| \leq \bar{\eta}R_{i,i} + \begin{vmatrix} 2^{-p+7}\kappa^2\phi(i_0)(\|\mathbf{b}_\kappa\| + R_{i_0,i_0}) & \text{if} \quad i \leq i_0 \\ 2^{-p}\phi(i)\|\mathbf{b}_\kappa\| & \text{if} \quad i > i_0. \end{vmatrix}$$

Proof. Suppose first that $i \leq i_0$. Then

$$\begin{aligned}|R''_{i,\kappa}| &= |R'_{i,\kappa} - X_i R_{i,i}| \\ &\leq \Delta R'_{i,\kappa} + |\overline{R}'_{i,\kappa} - X_i\overline{R}_{i,i}| + |X_i|\Delta R_{i,i} \\ &\leq \Delta R'_{i,\kappa} + \overline{R}_{i,i}\cdot\left|\frac{\overline{R}'_{i,\kappa}}{\overline{R}_{i,i}} - X_i\right| + |X_i|\Delta R_{i,i} \\ &\leq \Delta R'_{i,\kappa} + \frac{\overline{R}_{i,i}}{2} + 2^{-p}|\overline{R}'_{i,\kappa}| + \left(\frac{1}{2} + 2\frac{|\overline{R}'_{i,\kappa}|}{\overline{R}_{i,i}}\right)\Delta R_{i,i} \\ &\leq \Delta R'_{i,\kappa} + \frac{R_{i,i}}{2} + 2^{-p}|\overline{R}'_{i,\kappa}| + \left(1 + 2\frac{|\overline{R}'_{i,\kappa}|}{\overline{R}_{i,i}}\right)\Delta R_{i,i} \\ &\leq \Delta R'_{i,\kappa} + \left(\frac{1}{2} + 2^{-p}\phi(i)\right)R_{i,i} + 2^{-p+2}\phi(i)|\overline{R}'_{i,\kappa}|,\end{aligned}$$

where we used Corollary 2.4. Therefore, using Lemmata 3.4 and 3.5, we get the result.

Suppose now that $i > i_0$. Then, using Corollary 2.4:

$$\begin{aligned}|R''_{i,\kappa}| = |R'_{i,\kappa}| &\leq |\overline{R}'_{i,\kappa}| + \Delta R'_{i,\kappa} \\ &\leq \overline{R}_{i,i}/2 + 2^{-p}\phi(i)(\|\mathbf{b}_\kappa\| + R_{i,i}),\end{aligned}$$

which completes the proof. □

The latter bound on the $R''_{i,\kappa}$'s shows that at Step 6, the length of the vector $\mathbf{b}_\kappa$ is likely to decrease.

THEOREM 3.7. *Consider $\mathbf{b}_\kappa$ at the beginning of Step 6. Let $\mathbf{b}''_\kappa$ be its new value at the end of Step 6. Then*

$$\|\mathbf{b}''_\kappa\| \leq 2\kappa \max_{i\leq\kappa} R_{i,i} + 2^{-p+7}\kappa^3\phi(\kappa)\|\mathbf{b}_\kappa\|.$$

Proof. Using Lemma 3.6:

$$\begin{aligned}\|\mathbf{b}''_\kappa\| \leq \quad & \sum_{i=1}^{\kappa} |R''_{i,\kappa}| = R_{\kappa,\kappa} + \sum_{i=1}^{i_0} |R''_{i,\kappa}| + \sum_{i=i_0+1}^{\kappa-1} |R_{i,\kappa}| \\ \leq \quad & R_{\kappa,\kappa} + 2^{-p+7}\kappa^2 i_0\phi(i_0)R_{i_0,i_0} + \kappa\overline{\eta}\max_{i<\kappa} R_{i,i} \\ & + 2^{-p+7}\kappa^3\phi(\kappa)\|\mathbf{b}_\kappa\|.\end{aligned}$$

The latter provides the result. □

3.2 Correctness and Cost of Algorithm 3

The following lemma ensures the soundness of the test of Step 7. It also implies that the algorithm terminates.

LEMMA 3.8. *Consider $\mathbf{b}_\kappa$ at the beginning of Step 6. Let $\mathbf{b}''_\kappa$ be its new value at the end of Step 6. If the test of Step 7 succeeds, then $\|\mathbf{b}''_\kappa\|^2 \geq 2^{-cd-1}\|\mathbf{b}_\kappa\|^2$. If the test of Step 7 fails, then $\|\mathbf{b}''_\kappa\|^2 \leq 2^{-cd+1}\|\mathbf{b}_\kappa\|^2$.*

Proof. Using [4, Eq. (3.5)], we have for any $\mathbf{b} \in \mathbb{Z}^n$ that $\diamond(\|\mathbf{b}\|^2) \in (1 \pm n2^{-p+1})\|\mathbf{b}\|^2$. Thus $\diamond\left(\diamond(2^{-cd}) \cdot \diamond(\|\mathbf{b}_\kappa\|^2)\right) \in (1 \pm n2^{-p+2})2^{-cd}\|\mathbf{b}_\kappa\|^2$. □

The following shows that at the end of the execution of Algorithm 3, the length of $\mathbf{b}_\kappa$ and the $R_{i,\kappa}$'s are small. The algorithm is correct in the sense that the size of the output vector is bounded.

THEOREM 3.9. *Let $\overline{\theta} = 2^{-p+8+\frac{cd}{2}}\kappa^3\phi(\kappa)$ and $\overline{\eta} = 1/2 + 2^{-p+1}\phi(\kappa)$. At the end of the execution of Algorithm 3, we have:*

$$\begin{aligned}\|\mathbf{b}_\kappa\| \quad &\leq \quad 3\kappa \max_{i\leq\kappa} R_{i,i}, \\ \forall i < \kappa,\ |R_{i,\kappa}| \quad &\leq \quad \overline{\eta}R_{i,i} + \overline{\theta}(\|\mathbf{b}_\kappa\| + R_{\kappa-1,\kappa-1}).\end{aligned}$$

Proof. Lemma 3.8 gives us that $\|\mathbf{b}^\dagger_\kappa\|^2 \leq 2^{cd+1}\|\mathbf{b}_\kappa\|^2$, where $\mathbf{b}^\dagger_\kappa$ (resp. $\mathbf{b}_\kappa$) is the vector $\mathbf{b}_\kappa$ at the beginning (resp. at the end) of the last iteration of the loop made of Steps 1–7. Using Theorem 3.7, we obtain:

$$\begin{aligned}\|\mathbf{b}_\kappa\| &\leq 2\kappa \max_{i\leq\kappa} R_{i,i} + 2^{-p+7}\kappa^3\phi(\kappa)\|\mathbf{b}^\dagger_\kappa\| \\ &\leq 2\kappa \max_{i\leq\kappa} R_{i,i} + 2^{-p+8+\frac{cd}{2}}\kappa^3\phi(\kappa)\|\mathbf{b}^\dagger_\kappa\| \\ &\leq 3\kappa \max_{i\leq\kappa} R_{i,i}.\end{aligned}$$

For the second inequality, note that Lemma 3.6 implies:

$$|R_{i,\kappa}| \leq \overline{\eta}R_{i,i} + 2^{-p+7}\kappa^2\phi(\kappa)(\|\mathbf{b}^\dagger_\kappa\| + R_{\kappa-1,\kappa-1}).$$

It only remains to use the inequality $\|\mathbf{b}^\dagger_\kappa\|^2 \leq 2^{cd+1}\|\mathbf{b}_\kappa\|^2$. □

We now consider the cost of Algorithm 3. We start by bounding the number of iterations of the main loop.

LEMMA 3.10. *The number of iterations of the loop made of Steps 1–7 is:*

$$O\left(1 + \frac{1}{d}\log\frac{\|\mathbf{b}^b_\kappa\|}{\|\mathbf{b}^e_\kappa\|}\right),$$

where $\mathbf{b}^b_\kappa$ (resp. $\mathbf{b}^e_\kappa$) is $\mathbf{b}_\kappa$ at the beginning (resp. the end).

Proof. Let $\mathbf{b}^\ell_\kappa$ be the vector $\mathbf{b}_\kappa$ at the beginning of Step 2 of the last iteration of the loop made of Steps 1–7. Lemma 3.8 implies that the number of loop iterations is bounded by $1 + \frac{2}{cd-1}\log\frac{\|\mathbf{b}^b_\kappa\|}{\|\mathbf{b}^\ell_\kappa\|}$. If all the X_i's of the last iteration are zero, then $\mathbf{b}^e_\kappa = \mathbf{b}^\ell_\kappa$. Otherwise, since $X_{i_0} \neq 0$, Lemma 3.1 and Corollary 2.4 give:

$$\begin{aligned}\|\mathbf{b}^\ell_\kappa\| \geq |R^\ell_{i_0,\kappa}| &\geq |\overline{R}^\ell_{i_0,\kappa}| - \Delta R^\ell_{i_0,\kappa} \\ &\geq \frac{1}{4}|X_{i_0}|R_{i_0,i_0} - 2^{-p}\phi(i_0)(\|\mathbf{b}^\ell_\kappa\| + R_{i_0,i_0}) \\ &\geq \frac{1}{8}R_{i_0,i_0}.\end{aligned}$$

Furthermore, using Lemma 3.6, we get (noting $\mathbf{a} = (R^e_{1,\kappa}, \ldots, R^e_{i_0,\kappa}, 0, \ldots, 0)$ and $\mathbf{b} = (0, \ldots, 0, R^e_{i_0+1,\kappa}, \ldots, R^e_{\kappa,\kappa}, 0, \ldots, 0)$):

$$\begin{aligned}\|\mathbf{b}^e_\kappa\| - \|\mathbf{b}^\ell_\kappa\| &= \|\mathbf{r}^e_\kappa\| - \|\mathbf{r}^\ell_\kappa\| \\ &\leq \|\mathbf{a}\| + \|\mathbf{b}\| - \|\mathbf{b}\| \\ &\leq \sum_{i\leq i_0} |R^e_{i,\kappa}| \\ &\leq (\kappa\alpha^{i_0} + \overline{\theta})R_{i_0,i_0} + \overline{\theta}\|\mathbf{b}^\ell_\kappa\| \\ &\leq 9(\kappa\alpha^{i_0} + \overline{\theta})\|\mathbf{b}^\ell_\kappa\|.\end{aligned}$$

This gives that $\|\mathbf{b}^e_\kappa\| \leq 10\kappa\alpha^\kappa\|\mathbf{b}^\ell_\kappa\|$, which provides the bound. □

The result above leads us to the following complexity upper bound.

THEOREM 3.11. *Let $(\mathbf{b}_1, \ldots, \mathbf{b}_d) \in \mathbb{Z}^{n\times d}$ be a valid input to Algorithm 3. Let κ be the input index. Suppose the precision satisfies $p > \log_2(2^{\frac{cd}{2}+9}\kappa^3\phi(\kappa)\alpha/\theta)$ and $p = 2^{O(d)}$. Then the execution finishes within*

$$O\left[\left(d + \log\frac{\|\mathbf{b}^b_\kappa\|}{\|\mathbf{b}^e_\kappa\|}\right)\frac{n\mathcal{M}(d)}{d}(d + \log\|B\|)\right] \text{ bit operations,}$$

where $\|B\| = \max_{i\leq\kappa}\|\mathbf{b}_i\|$ and $\mathbf{b}^b_\kappa$ (resp. $\mathbf{b}^e_\kappa$) is $\mathbf{b}_\kappa$ at the beginning of Step 1 (resp. Step 9).

Proof. The bit-cost of one iteration of Steps 4 and 5 is $O(d\mathcal{M}(d))$ for handling the mantissas (thanks to the second restriction on p) and $O(d\log(d + \log\|B\|))$ for handling the exponents (thanks to Corollary 2.4 and Lemmata 3.1 and 3.4). This implies that one iteration of the loop made of Steps 3–5 costs $O(d^2\mathcal{M}(d) + d^2\log\log\|B\|)$. A similar bound $O(nd\mathcal{M}(d) + nd\log\log\|B\|)$ holds for one iteration of Step 2. The computation of t at Step 6 is negligible compared to the costs above. Theorem 3.9 implies that the update of $\mathbf{b}_\kappa$ at Step 6 can be performed within $O(n\mathcal{M}(d)\log(d\|B\|))$ bit operations (note that though X_i can be a very large integer, it is stored on $\leq p = O(d)$ bits). The cost of Step 7 is also negligible compared to the costs above. Overall, the bit-cost of one iteration of the loop consisting of Steps 1–7 is $O(n\mathcal{M}(d)(d + \log\|B\|))$. Lemma 3.10 provides the result. □

4. AN LLL RELYING ON HOUSEHOLDER'S ALGORITHM

The H-LLL algorithm (Algorithm 4) follows the general structure of LLL algorithms (see Algorithm 1). For the size-reduction, it relies on Algorithm 3. The precision requirement is a little stronger than in the previous section. Asymptotically, for close to optimal parameters δ, η and θ (i.e., $\delta \approx 1$, $\eta \approx 1/2$ and $\theta \approx 0$), a sufficient precision is $p \approx d$.

Algorithm 4 The H-LLL algorithm.

Input: A matrix $B \in \mathbb{Z}^{n\times d}$ of rank d and valid LLL parameters δ, η and θ, with $\theta < \eta - 1/2$.
Input: $\diamond(2^{-cd})$ (for an arbitrary $c > 0$) and a fp precision $p > p_0 + 1 - \log_2(1-\delta) - \log_2(\eta - \theta - 1/2)$ with $p_0 := \log_2(d^3\phi(d)\alpha^d/\theta) + 16 + cd/2$.
Output: A (δ, η, θ)-LLL-reduced basis of the lattice spanned by the columns of B.
1: Let $\overline{\delta}$ be a fp number in $(\delta + 2^{-p+p_0}, 1 - 2^{-p+p_0})$.
2: Compute $\overline{\mathbf{r}}_1, \overline{\mathbf{v}}_1, \sigma_1$ using Steps 3–9 of Algorithm 2.
3: $\kappa := 2$. While $\kappa \le d$, do
4: Call Algorithm 3 on $B, \overline{\mathbf{r}}_1, \ldots, \overline{\mathbf{r}}_{\kappa-1}, \overline{\mathbf{v}}_1, \ldots, \overline{\mathbf{v}}_{\kappa-1}$ and $\sigma_1, \ldots, \sigma_{\kappa-1}$.
5: $s := \diamond(\| \diamond (\mathbf{b}_\kappa)\|^2)$; $s := \diamond(s - \sum_{i\le\kappa-2} \overline{R}^2_{i,\kappa})$.
6: If $\diamond(\overline{\delta} \cdot \diamond(\overline{R}^2_{\kappa-1,\kappa-1})) \le s$, then $\kappa := \kappa + 1$.
7: Else swap $\mathbf{b}_{\kappa-1}$ and $\mathbf{b}_\kappa$; $\kappa := \max(\kappa - 1, 2)$.
8: Return B.

Before proceeding to the analysis of Algorithm 4, let us explain how Step 5 is performed. We compute $\diamond(\| \diamond (\mathbf{b}_\kappa)\|^2)$ sequentially; we compute the $\diamond(\overline{R}^2_{i,\kappa})$'s; and finally we compute $s := \diamond(\| \diamond (\mathbf{b}_\kappa)\|^2 - \sum_{i\le\kappa-2} \overline{R}^2_{i,\kappa})$ sequentially. Corollary 2.4 and Theorem 3.9 provide the soundness of such a computation.

LEMMA 4.1. *Assume that the first $\kappa-1$ columns of B are LLL-reduced. Then at the end of Step 5, we have:*

$$\left|s - (R^2_{\kappa,\kappa} + R^2_{\kappa-1,\kappa})\right| \le 2^{-p+12}\kappa^3\alpha^\kappa\phi(\kappa)(R^2_{\kappa,\kappa} + R^2_{\kappa-1,\kappa-1}).$$

Proof. First of all, thanks to [4, Eq. (3.5)], we have $|\diamond\|\diamond(\mathbf{b}_\kappa)\|^2 - \|\mathbf{b}_\kappa\|^2| \le n2^{-p+1}\|\mathbf{b}_\kappa\|^2$. Also:

$$\begin{aligned}|\diamond(\overline{R}^2_{i,\kappa}) - R^2_{i,\kappa}| &\le 2^{-p+1}R^2_{i,\kappa} + 2|\overline{R}^2_{i,\kappa} - R^2_{i,\kappa}| \\ &\le 2^{-p+1}R^2_{i,\kappa} + 2\Delta R_{i,\kappa}(|R_{i,\kappa}| + \Delta R_{i,\kappa}).\end{aligned}$$

Thanks to the LLL-reducedness of the first $\kappa-1$ columns of B, Corollary 2.4 and Theorem 3.9, we have (using $\overline{\theta} \le \alpha^{-\kappa}$):

$$\begin{aligned}|R_{i,\kappa}| &\le 2(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + \alpha^{-\kappa}\|\mathbf{b}_\kappa\|) \\ &\le 8\kappa(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + R_{\kappa,\kappa}) \\ \Delta R_{i,\kappa} &\le 2^{-p}\phi(i)(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + \|\mathbf{b}_\kappa\|) \\ &\le 2^{-p+2}\kappa\phi(i)(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + R_{\kappa,\kappa}).\end{aligned}$$

As a consequence, we obtain the bound:

$$\begin{aligned}|\diamond(\overline{R}^2_{i,\kappa}) - R^2_{i,\kappa}| &\le 2^{-p+8}\kappa^2\alpha^{2\kappa}(R^2_{\kappa-1,\kappa-1} + R^2_{\kappa,\kappa}) \\ &+2^{-p+7}\kappa^2\phi(i)(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + R_{\kappa,\kappa})^2 \\ &\le 2^{-p+9}\kappa^2\alpha^\kappa\phi(\kappa)(R^2_{\kappa-1,\kappa-1} + R^2_{\kappa,\kappa}).\end{aligned}$$

Finally, using [4, Eq. (3.5)], we get the bound:

$$\begin{aligned}|s - (R^2_{\kappa,\kappa} + R^2_{\kappa-1,\kappa-1})| &\le \kappa 2^{-p+1}(R^2_{\kappa,\kappa} + R^2_{\kappa-1,\kappa-1}) \\ &+ 2|\diamond\|\mathbf{b}_\kappa\|^2 - \|\mathbf{b}_\kappa\|^2| + 2\sum_{i\le\kappa-2}|\diamond(\overline{R}^2_{i,\kappa}) - R^2_{i,\kappa}|,\end{aligned}$$

which leads to the result. □

LEMMA 4.2. *Assume that the first $\kappa-1$ columns of B are LLL-reduced. Then at the end of Step 5, we have:*

$$|\diamond(\overline{\delta} \cdot \diamond(\bar{R}^2_{\kappa-1,\kappa-1})) - \bar{\delta}\bar{R}^2_{\kappa-1,\kappa-1}| \le 2^{-p+3}\phi(\kappa)\bar{\delta}\bar{R}^2_{\kappa-1,\kappa-1}.$$

Lemmata 4.1 and 4.2 imply the soundness of the test of Step 6.

THEOREM 4.3. *Let $\overline{\theta} = 2^{-p+8+\frac{cd}{2}}d^3\phi(d)$ and $\bar{\eta} = 1/2 + 2^{-p+1}\phi(d)$. Assume that the first $\kappa-1$ columns of B are (δ,η,θ)-LLL-reduced. If the test of Step 6 succeeds then the first κ columns of B are (δ,η,θ)-LLL-reduced. Otherwise $\delta' R^2_{\kappa-1,\kappa-1} > R^2_{\kappa,\kappa} + R^2_{\kappa-1,\kappa}$ with $\delta' = \overline{\delta}(1 + 2^{-p+14}\kappa^3\phi(\kappa)\alpha^\kappa)$.*

Proof. Suppose that the test succeeds. Corollary 2.4 and Lemmata 4.1 and 4.2 imply:

$$\begin{aligned}&(1 - 2^{-p+3}\phi(\kappa))\overline{\delta}R^2_{\kappa-1,\kappa-1} \\ &\quad\le (1 + 2^{-p+12}\kappa^3\alpha^\kappa\phi(\kappa))(R^2_{\kappa,\kappa} + R^2_{\kappa-1,\kappa-1}).\end{aligned}$$

By choice of $\overline{\delta}$, this implies that $\delta R^2_{\kappa-1,\kappa-1} \le R^2_{\kappa-1,\kappa} + R^2_{\kappa,\kappa}$.

Now, using Theorem 3.9, we know that:

$$\begin{aligned}|R_{\kappa-1,\kappa}| &\le (\overline{\eta} + \overline{\theta})R_{\kappa-1,\kappa-1} + \overline{\theta}\|\mathbf{b}_\kappa\| \\ &\le (\overline{\eta} + \overline{\theta}(1 + 3\kappa\alpha^\kappa))R_{\kappa-1,\kappa-1} + 3\overline{\theta}\kappa R_{\kappa,\kappa} \\ &\le \eta R_{\kappa-1,\kappa-1} + \theta R_{\kappa,\kappa}.\end{aligned}$$

As a consequence, we have $R_{\kappa-1,\kappa-1} \le \alpha R_{\kappa,\kappa}$. By using Theorem 3.9 again, we have:

$$\begin{aligned}|R_{i,\kappa}| &\le \overline{\eta}R_{i,i} + \overline{\theta}(\|\mathbf{b}_\kappa\| + R_{\kappa-1,\kappa-1}) \\ &\le \overline{\eta}R_{i,i} + \overline{\theta}(3\kappa\max_{j\le\kappa}(R_{j,j}) + \alpha R_{\kappa,\kappa}) \\ &\le \overline{\eta}R_{i,i} + 4\overline{\theta}\kappa\alpha^\kappa R_{\kappa,\kappa},\end{aligned}$$

which completes the proof of the first claim of the theorem.

Suppose now that the test fails. Corollary 2.4 and Lemmata 4.1 and 4.2 imply:

$$\begin{aligned}&(1 + 2^{-p+3}\phi(\kappa))\overline{\delta}R^2_{\kappa-1,\kappa-1} \\ &\quad\ge (1 - 2^{-p+12}\kappa^3\alpha^\kappa\phi(\kappa))(R^2_{\kappa,\kappa} + R^2_{\kappa-1,\kappa-1}).\end{aligned}$$

By definition of δ', this implies that $\delta' R^2_{\kappa-1,\kappa-1} > R^2_{\kappa-1,\kappa} + R^2_{\kappa,\kappa}$. □

We can now conclude our study of Algorithm 4.

THEOREM 4.4. *Algorithm 4 returns a (δ,η,θ)-LLL-reduced basis $(\mathbf{b}^e_1, \ldots, \mathbf{b}^e_d)$ of the lattice spanned by the input basis $(\mathbf{b}^b_1, \ldots, \mathbf{b}^b_d) \in \mathbb{Z}^{n\times d}$. Furthermore, the bit complexity is*

$$O\left[\left(d + \log\prod\frac{d^b_i}{d^e_i} + \frac{1}{d}\log\prod\frac{\|\mathbf{b}^b_i\|}{\|\mathbf{b}^e_i\|}\right)n\mathcal{M}(d)(d + \log\|B\|)\right],$$

where $\|B\| = \max\|\mathbf{b}_i\|$ and d^b_i (resp. d^e_i) is the determinant of the lattice spanned by the first i columns of the input (resp. output) basis. The complexity bound above is itself $O(nd^2\mathcal{M}(d)\log\|B\|(d + \log\|B\|))$.

Proof. Using the classical analysis of the LLL algorithm [7] and Theorem 4.3, we know that the algorithm terminates within $O\left(d + \log \prod_{i \leq d} \frac{d_i^b}{d_i^e}\right)$ iterations. A simple induction using Theorem 4.3 proves that the output is indeed (δ, η, θ)-LLL reduced. Furthermore, the classical analysis of LLL yields that at any moment, the norms of the basis vectors are below $d\|B\|$ (except within the calls to Algorithm 3). Each call to Algorithm 3 that transforms $\mathbf{b}_\kappa^{(old)}$ into $\mathbf{b}_\kappa^{(new)}$ costs

$$O\left[\left(d + \log \frac{\|\mathbf{b}_\kappa^{(old)}\|}{\|\mathbf{b}_\kappa^{(new)}\|}\right) \frac{n\mathcal{M}(d)}{d} (d + \log \|B\|)\right] \text{ bit operations.}$$

As a consequence, the total cost of Algorithm 4 is (using the fact that the product over the loop iterations of the $\frac{\|\mathbf{b}_\kappa^{(old)}\|}{\|\mathbf{b}_\kappa^{(new)}\|}$'s is exactly $\prod_i \frac{\|\mathbf{b}_i^b\|}{\|\mathbf{b}_i^e\|}$):

$$\begin{aligned} &O\left[\sum_{\text{iterations}} \left(d + \log \frac{\|\mathbf{b}_\kappa^{(old)}\|}{\|\mathbf{b}_\kappa^{(new)}\|}\right) \frac{n\mathcal{M}(d)}{d} (d + \log \|B\|)\right] \\ =&O\left[\left(d + \log \prod \frac{d_i^b}{d_i^e} + \frac{1}{d} \log \prod \frac{\|\mathbf{b}_i^b\|}{\|\mathbf{b}_i^e\|}\right) n\mathcal{M}(d)(d + \log \|B\|)\right]. \end{aligned}$$

Since $\prod \|\mathbf{b}_i^b\| \leq \|B\|^d$ and $\prod d_i^b \leq \|B\|^{d^2}$, that bound immediately gives a $O(nd^2\mathcal{M}(d) \log \|B\|(d + \log \|B\|))$ complexity upper bound. □

5. CONCLUSION

The decision to use Householder's transformations instead of Cholesky's factorization within LLL leads to modifications in the proof of correctness: the perturbations induced on the approximate R-factor have a different structure than in the L^2 algorithm of [9]. These modifications may probably be used for other forms or applications of the floating-point reduction of lattices. For example the new approach may be carried over to the case of linearly dependent input vectors, and to the case of stronger reductions (such as the fp Hermite-Korkine-Zolotarev reduction algorithm of [11]). An important direction that deserves to be investigated would be to try to further decrease the precision of the approximate computations. We showed that a precision essentially equal to the problem dimension is sufficient. Can we do better? It seems unnatural that a higher precision is required in H-LLL than in its (incomplete) underlying size-reduction algorithm. Finally, a more precise understanding of the numerical behavior is required for various aspects, such as the efficient implementation of H-LLL, which we are currently investigating.

ACKNOWLEDGMENTS. We thank the anonymous referees for their helpful comments. Ivan Morel and Damien Stehlé were partly funded by the LaRedA ANR project. Gilles Villard was partly funded by the Gecko ANR project.

6. REFERENCES

[1] X.-W. Chang, D. Stehlé, and G. Villard. Perturbation Analysis of the R-Factor of the QR Factorisation in the Context of LLL-Reduction. Work in progress, available at `http://perso.ens-lyon.fr/damien.stehle/QRPERTURB.html`, 2009.

[2] H. Cohen. *A Course in Computational Algebraic Number Theory, 2nd edition*. Springer, 1995.

[3] C. Hermite. Extraits de lettres de M. Hermite à M. Jacobi sur différents objets de la théorie des nombres, deuxième lettre. *J. reine angew Math*, 40:279–290, 1850.

[4] N. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.

[5] E. Kaltofen. On the complexity of finding short vectors in integer lattices. In *Proc. of EUROCAL'83*, volume 162 of *LNCS*, pages 236–244. Springer, 1983.

[6] H. Koy and C. P. Schnorr. Segment LLL-reduction of lattice bases with floating-point orthogonalization. In *Proc. of CALC'01*, volume 2146 of *LNCS*, pages 81–96. Springer, 2001.

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

[8] L. Lovász. *An Algorithmic Theory of Numbers, Graphs and Convexity*. SIAM, 1986. CBMS-NSF Regional Conference Series in Applied Mathematics.

[9] P. Nguyen and D. Stehlé. Floating-point LLL revisited. In *Proc. of Eurocrypt 2005*, volume 3494 of *LNCS*, pages 215–233. Springer, 2005. Extended version to appear in SIAM J. Comput., 2009.

[10] A. M. Odlyzko. The rise and fall of knapsack cryptosystems. In *Proc. of Cryptology and Computational Number Theory*, volume 42 of *Proc. of Symposia in Applied Mathematics*, pages 75–88. AMS, 1989.

[11] X. Pujol and D. Stehlé. Rigorous and efficient short lattice vectors enumeration. In *Proc. of Asiacrypt'08*, volume 5350 of *LNCS*, pages 390–405. Springer, 2008.

[12] C. P. Schnorr. Progress on LLL and lattice reduction. In *Proc. of the LLL+25 conference*. To appear in 2009.

[13] C. P. Schnorr. A more efficient algorithm for lattice basis reduction. *J. of Alg.*, 9(1):47–62, 1988.

[14] C. P. Schnorr. Fast LLL-type lattice reduction. *Inf. and Comp*, 204:1–25, 2006.

[15] C. P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Math. of Prog*, 66:181–199, 1994.

[16] D. Stehlé. Floating-point LLL: theoretical and practical aspects. In *Proc. of the LLL+25 conference*. To appear in 2009.

[17] A. Storjohann. Faster Algorithms for Integer Lattice Basis Reduction. Technical Report TR249, ETH-Zurich, Dpt. Comp. Sc., 1996.

[18] G. Villard. Certification of the QR factor R, and of lattice basis reducedness. In *Proc. ISSAC '07*, pages 361–368. ACM Press, 2007.

An LLL-Reduction Algorithm with Quasi-linear Time Complexity[1]

Andrew Novocin, Damien Stehlé, and Gilles Villard

CNRS, ENS de Lyon, INRIA, UCBL, U. Lyon
Laboratoire LIP 46 Allée d'Italie, 69364 Lyon Cedex 07, France.
{andrew.novocin,damien.stehle,gilles.villard}@ens-lyon.fr

Abstract. We devise an algorithm, $\widetilde{\mathrm{L}}^1$, with the following specifications: It takes as input an arbitrary basis $B = (\mathbf{b}_i)_i \in \mathbb{Z}^{d\times d}$ of a Euclidean lattice L; It computes a basis of L which is reduced for a mild modification of the Lenstra-Lenstra-Lovász reduction; It terminates in time $O(d^{5+\varepsilon}\beta + d^{\omega+1+\varepsilon}\beta^{1+\varepsilon})$ where $\beta = \log \max \|\mathbf{b}_i\|$ (for any $\varepsilon > 0$ and ω is a valid exponent for matrix multiplication). This is the first LLL-reducing algorithm with a time complexity that is quasi-linear in β and polynomial in d.
The backbone structure of $\widetilde{\mathrm{L}}^1$ is able to mimic the Knuth-Schönhage fast gcd algorithm thanks to a combination of cutting-edge ingredients. First the bit-size of our lattice bases can be decreased via truncations whose validity are backed by recent numerical stability results on the QR matrix factorization. Also we establish a new framework for analyzing unimodular transformation matrices which reduce shifts of reduced bases, this includes bit-size control and new perturbation tools. We illustrate the power of this framework by generating a family of reduction algorithms.

1 Introduction

We present the first lattice reduction algorithm which has complexity both quasi-linear in the bit-length of the entries and polynomial time overall for an input basis $B = (\mathbf{b}_i)_i \in \mathbb{Z}^{d\times d}$. This is the first progress on quasi-linear lattice reduction in nearly 10 years, improving Schönhage [28], Yap [32], and Eisenbrand and Rote [7] whose algorithm is exponential in d. Our result can be seen as a generalization of the Knuth-Schönhage quasi-linear GCD [13, 26] from integers to matrices. For solving the matrix case difficulties which relate to multi-dimensionality we combine several new main ingredients. We establish a theoretical framework for analyzing and designing general lattice reduction algorithms. In particular we discover an underlying structure on any transformation matrix which reduces shifts of reduced lattices; this new structure reveals some of the inefficiencies of traditional lattice reduction algorithms. The multi-dimensional difficulty also leads us to establish new perturbation analysis results for mastering the complexity bounds. The Knuth-Schönhage scalar approach essentially relies on truncations of the Euclidean remainders [13, 26] , while the matrix case requires truncating both the "remainder" and "quotient" matrices. We can use our theoretical framework to propose a family of new reduction algorithms, which includes a Lehmer-type sub-quadratic algorithm in addition to $\widetilde{\mathrm{L}}^1$.

In 1982, Lenstra, Lenstra and Lovász devised an algorithm, L^3, that computes reduced bases of integral Euclidean lattices (i.e., subgroups of a $\mathbb{Z}^d$) in polynomial time [16]. This typically allows one to solve approximate variants of computationally hard problems such as the Shortest Vector, Closest Vector, and the Shortest Independent Vectors problems (see [18]). L^3 has since proven useful in dozens of applications in a wide range including cryptanalysis, computer algebra, communications theory, combinatorial optimization, algorithmic number theory, etc (see [22, 6] for two recent surveys).

[1] Extended abstract appears in the *Proc. 43rd ACM Symposium on Theory of Computing (STOC 2011), June 6-8, San Jose, California, 2011.*

In [16], Lenstra, Lenstra and Lovász bounded the bit-complexity of L^3 by $\mathcal{O}(d^{5+\varepsilon}\beta^{2+\varepsilon})$ when the input basis $B = (\mathbf{b}_i)_i \in \mathbb{Z}^{d\times d}$ satisfies $\max \|\mathbf{b}_i\| \leq 2^\beta$. For the sake of simplicity, we will only consider full-rank lattices. The current best algorithm for integer multiplication is Fürer's, which allows one to multiply two k-bit long integers in time $\mathcal{M}(k) = \mathcal{O}(k(\log k)2^{\log^* k})$. The analysis of L^3 was quickly refined by Kaltofen [11], who showed a $\mathcal{O}(d^5\beta^2(d+\beta)^\varepsilon)$ complexity bound. Schnorr [24] later proposed an algorithm of bit-complexity $\mathcal{O}(d^4\beta(d+\beta)^{1+\varepsilon})$, using approximate computations for internal Gram-Schmidt orthogonalizations. Some works have since focused on improving the complexity bounds with respect to the dimension d, including [27, 30, 14, 25], but they have not lowered the cost with respect to β (for fixed d). More recently, Nguyen and Stehlé devised L^2 [21], a variant of L^3 with complexity $\mathcal{O}(d^{4+\varepsilon}\beta(d+\beta))$. The latter bound is quadratic with respect to β (even with naive integer multiplication), which led to the name L^2. The same complexity bound was also obtained in [20] for a different algorithm, H-LLL, but with a simpler complexity analysis.

As a broad approximation, L^3, L^2 and H-LLL are generalizations of Euclid's greatest common divisor algorithm. The successive bases computed during the execution play the role of Euclid's remainders, and the elementary matrix operations performed on the bases play the role of Euclid's quotients. L^3 may be interpreted in such a framework. It is slow because it computes its "quotients" using all the bits from the "remainders" rather than the most significant bits: The cost of computing one Euclidean division in an L^3 way is $\mathcal{O}(\beta^{1+\varepsilon})$, leading to an overall $\mathcal{O}(\beta^{2+\varepsilon})$ bound for Euclid's algorithm. Lehmer [15] proposed an acceleration of Euclid's algorithm by the means of truncations. Since the ℓ most significant bits of the remainders provide the first $\Omega(\ell)$ bits of the sequence of quotients, one may: Truncate the remainders to precision ℓ; Compute the sequence of quotients for the truncated remainders; Store the first $\Omega(\ell)$ bits of the quotients into an $\Omega(\ell)$-bit matrix; Apply the latter to the input remainders, which are shortened by $\Omega(\ell)$ bits; And iterate. The cost gain stems from the decrease of the bit-lengths of the computed remainders. Choosing $\ell \approx \sqrt{\beta}$ leads to a complexity bound of $\mathcal{O}(\beta^{3/2+\varepsilon})$. In the early 70's, Knuth [13] and Schönhage [26] independently observed that using Lehmer's idea recursively leads to a gcd algorithm with complexity bound $\mathcal{O}(\beta^{1+\varepsilon})$. The above approach for the computation of gcds has been successfully adapted to two-dimensional lattices [32, 28, 5], and the resulting algorithm was then used in [7] to reduce lattices in arbitrary dimensions in quasi-linear time. Unfortunately, the best known cost bound for the latter is $\mathcal{O}(\beta^{1+\varepsilon}(\log\beta)^{d-1})$ for fixed d.

Our result. We adapt the Lehmer-Knuth-Schönhage gcd framework to the case of LLL-reduction. $\widetilde{\mathrm{L}}^1$ takes as input a non-singular $B \in \mathbb{Z}^{d\times d}$; terminates within $\mathcal{O}(d^{5+\varepsilon}\beta + d^{\omega+1+\varepsilon}\beta^{1+\varepsilon})$ bit operations, where $\beta = \log\max\|\mathbf{b}_i\|$; and returns a basis of the lattice $L(B)$ spanned by B which is LLL-reduced in the sense of Definition 1 given hereafter. (L^3 reduces bases for $\Xi = (3/4, 1/2, 0)$.) The time bound is obtained via an algorithm that can multiply two $d \times d$ matrices in $\mathcal{O}(d^\omega)$ scalar operations. (We can set $\omega \approx 2.376$ [4].) Our complexity improvement is particularly relevant for applications of LLL reduction where β is large. These include the recognition of algebraic numbers [12] and Coppersmith's method for finding the small roots of polynomials [3].

Definition 1 ([2, Def. 5.3]). *Let $\Xi = (\delta, \eta, \theta)$ with $\eta \in (1/2, 1)$, $\theta > 0$ and $\delta \in (\eta^2, 1)$. Let $B \in \mathbb{R}^{d\times d}$ be non-singular with QR factorization $B = Q \cdot R$ (i.e., the unique decomposition of B as a product of an orthogonal matrix and an upper triangular matrix with positive diagonal entries). The matrix B is Ξ-LLL-reduced if:*

- *for all $i < j$, we have $|r_{i,j}| \leq \eta r_{i,i} + \theta r_{j,j}$ (B is size-reduced);*
- *for all i, we have $\delta \cdot r_{i,i}^2 \leq r_{i,i+1}^2 + r_{i+1,i+1}^2$ (B is said to satisfy Lovász' conditions).*

Let $\Xi_i = (\delta_i, \eta_i, \theta_i)$ be valid LLL-parameters for $i \in \{1, 2\}$. We say that Ξ_1 is stronger than Ξ_2 and write $\Xi_1 > \Xi_2$ if $\delta_1 > \delta_2$, $\eta_1 < \eta_2$ and $\theta_1 < \theta_2$.

This modified LLL-reduction is as powerful as the classical one (note that by choosing (δ, η, θ) close to the ideal parameters $(1, 1/2, 0)$, the derived α tends to $2/\sqrt{3}$):

Theorem 1 ([2, Th. 5.4]). *Let $B \in \mathbb{R}^{d\times d}$ be (δ, η, θ)-LLL-reduced with R-factor R. Let $\alpha = \frac{\eta\theta+\sqrt{(1+\theta^2)\delta-\eta^2}}{\delta-\eta^2}$. Then, for all i, $r_{i,i} \leq \alpha \cdot r_{i+1,i+1}$ and $r_{i,i} \leq \|\mathbf{b}_i\| \leq \alpha^i \cdot r_{i,i}$. This implies that $\|\mathbf{b}_1\| \leq \alpha^{\frac{d-1}{2}} |\det B|^{1/d}$ and $\alpha^{i-d} r_{i,i} \leq \lambda_i \leq \alpha^i r_{i,i}$, where λ_i is the ith minimum of the lattice $L(B)$.*

$\widetilde{\mathrm{L}}^1$ and its analysis rely on two recent lattice reduction techniques (described below), whose contributions can be easily explained in the gcd framework. The efficiency of the fast gcd algorithms [13, 26] stems from two sources: Performing operations on truncated remainders is meaningful (which allows one to consider remainders with smaller bit-sizes), and the obtained transformations corresponding to the quotients sequence have small bit-sizes (which allows one to transmit at low cost the information obtained on the truncated remainders back to the genuine remainders). We achieve an analogue of the latter by *gradually feeding the input* to the reduction algorithm, and the former is ensured thanks to the modified notion of LLL-reduction which is *resilient to truncations*.

The main difficulty in adapting the fast gcd framework lies in the multi-dimensionality of lattice reduction. In particular, the basis vectors may have significantly differing magnitudes. This means that basis truncations must be performed vector-wise. (Column-wise using the matrix setting.) Also, the resulting unimodular transformation matrices (integral with determinant ± 1 so that the spanned lattice is preserved) may have large magnitudes, hence need to be truncated for being be stored on few bits.

To solve these dilemmas we focus on reducing bases which are a mere scalar shift from being reduced. We call this process *lift-reducing*, and it can be used to provide a family of new reduction algorithms. We illustrate in Section 2 that the general lattice reduction problem can be reduced to the problem of lift-reduction. Indeed, the LLL-reduction of B can be implemented as a sequence of lift-reductions by performing a Hermite Normal Form (HNF) computation on B beforehand. Note that there could be other means of seeding the lift-reduction process. Our lift-reductions are a generalization of recent gradual feeding algorithms.

Gradual feeding of the input. Gradual feeding was introduced by Belabas [1], Novocin, and van Hoeij [23, 10], in the context of specific lattice bases that are encountered while factoring rational polynomials (e.g., with the algorithm from [9]). Gradual feeding was restricted to reducing specific sub-lattices which avoid the above dimensionality difficulties. We generalize these results to the following. Suppose that we wish to reduce a matrix B with the property that $B_0 := \sigma_\ell^{-k} B$ is reduced for some k and σ_ℓ is the diagonal matrix $\mathrm{diag}(2^\ell, 1, \ldots, 1)$. If one runs L^3 on B directly then the structure of B_0 is not being exploited. Instead, the matrix B can be slowly reduced allowing us to control and understand the intermediate transformations: Compute the unimodular transform U_1 (with any reduction algorithm) such that $\sigma_\ell B_0 U_1$ is reduced and repeat until we have $\sigma_\ell^k B_0 U_1 \cdots U_k = B(U_1 \cdots U_k)$. Each entry of U_i and each entry of $U_1 \cdots U_i$ can be bounded sensitive to the shape of the lattice. Further we will illustrate that the bit-size of any entry of U_i can be made $\mathcal{O}(\ell + d)$ (see Theorems 2 and 4).

In addition, control over U gives us the ability to analyze the impact of efficient truncations on lift-reductions.

Truncations of basis matrices. In order to work on as few bits of basis matrices as possible during our lift-reductions, we apply column-wise truncations. A truncation of precision p replaces a matrix B by a truncated matrix $B + \Delta B$ such that $\max \frac{\|\Delta \mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-p}$ holds for all i, and only the most significant $p + \mathcal{O}(\log d)$ bits of every column of $B + \Delta B$ are allowed to be non-zero. Each entry of $B + \Delta B$ is an integer multiplied by some power of 2. (In the notation ΔB, Δ does not represent anything, i.e., the matrix ΔB is not a product of Δ and B.) A truncation is an efficiency-motivated column-wise perturbation. The following lemmata explain why we are interested in such perturbations.

Lemma 1 ([2, Se. 2], refined from [8]). *Let $p > 0$, $B \in \mathbb{R}^{d\times d}$ non-singular with R-factor R, and let ΔB with $\max \frac{\|\Delta \mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-p}$. If $\mathrm{cond}(R) = \||R||R^{-1}|\|_2$ (using the induced norm) satisfies $c_0 \cdot \mathrm{cond}(R) \cdot 2^{-p} < 1$ with $c_0 = 8d^{3/2}$, then $B + \Delta B$ is non-singular and its R-factor $R + \Delta R$ satisfies $\max \frac{\|\Delta \mathbf{r}_i\|}{\|\mathbf{r}_i\|} \leq c_0 \cdot \mathrm{cond}(R) \cdot 2^{-p}$.*

Lemma 2 ([2, Le. 5.5]). *If $B \in \mathbb{R}^{d\times d}$ with R-factor R is (δ, η, θ)-reduced then $\mathrm{cond}(R) \leq \frac{\rho+1}{\rho-1}\rho^d$, with $\rho = (1 + \eta + \theta)\alpha$, with α as in Theorem 1.*

These results imply that a column-wise truncation of a reduced basis with precision $\Omega(d)$ remains reduced. This explains why the parameter θ was introduced in Definition 1, as such a property does not hold if LLL-reduction is restricted to $\theta = 0$ (see [29, Se. 3.1]).

Lemma 3 ([2, Co. 5.1]). *Let $\Xi_1 > \Xi_2$ be valid reduction parameters. There exists a constant c_1 such that for any Ξ_1-reduced $B \in \mathbb{R}^{d\times d}$ and any ΔB with $\max \frac{\|\Delta \mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-c_1 \cdot d}$, the matrix $B + \Delta B$ is non-singular and Ξ_2-reduced.*

As we will see in Section 3 (see Lemma 7) the latter lemmata will allow us to develop the gradual reduction strategy with truncation, which is to approximate the matrix to be reduced, reduce that approximation, and apply the unimodular transform to the original matrix, and repeat the process.

Lift-$\widetilde{\mathrm{L}}^1$. Our quasi-linear general lattice reduction algorithm, $\widetilde{\mathrm{L}}^1$, is composed of a sequence of calls to a specialized lift-reduction algorithm, Lift-$\widetilde{\mathrm{L}}^1$. Sections 2 and 4.4 show the relationship between general reduction and lift-reduction via HNF.

Inputs: B_0 reduced, and target lift ℓ.
Output: U_{small} such that $\sigma_\ell B_0 U_{\text{small}}$ is reduced.

1. Get $U_{1,\text{small}}$ from `pseudo-Lift-`$\widetilde{\mathrm{L}}^1$`(truncate`$(B_0), \ell/2)$.
2. $B_1 := \sigma_{\ell/2} B_0 U_{1,\text{small}}$.
3. Get U from `refineReduction`(C).
4. Get $U_{2,\text{small}}$ from `pseudo-Lift-`$\widetilde{\mathrm{L}}^1$`(truncate`$(B_1 U), \ell/2)$.
5. $U_{\text{small}} :=$`clean`$(U_{1,\text{small}} \cdot U \cdot U_{2,\text{small}})$.
6. Return U_{small}.

Fig. 1. `pseudo-Lift-`$\widetilde{\mathrm{L}}^1$.

When we combine lift-reduction (gradual feeding) and truncation we see another difficulty which must be addressed. That is, lift-reducing a truncation of B_0 will not give the same transformation as lift-reducing B_0 directly; likewise any truncation of U weakens our reduction even further. Thus after working with truncations we must apply any transformations to a higher

precision lattice and refine the result. In other words, we will need to have a method for strengthening the quality of a weakly reduced basis. Such an algorithm exists in [19] and we adapt it to performing lift-reductions in section 3.2. Small lift-reductions with this algorithm also become the leaves of our recursive tree. The Lift-$\widetilde{\mathrm{L}}^1$ algorithm in Figure 4 is a rigorous implementation of the pseudo algorithm in Figure 1: Lift-$\widetilde{\mathrm{L}}^1$ must refine current matrices more often than this pseudo algorithm to properly handle a specified reduction.

It could be noted that `clean` is stronger than mere truncation. It can utilize our new understanding of the structure of any lift-reducing U to provide an appropriate transformation which is well structured and efficiently stored.

Comments on the cost of $\widetilde{\mathrm{L}}^1$. The term $\mathcal{O}(d^{5+\varepsilon}\beta)$ stems from a series of β calls to H-LLL [20] or L^2 [21] on integral matrices whose entries have bit-lengths $\mathcal{O}(d)$. These calls are at the leaves of the tree of the recursive algorithm. An amortized analysis allows us to show that the total number of LLL switches performed summed over all calls is $\mathcal{O}(d^2\beta)$ (see Lemma 11). We recall that known LLL reduction algorithms perform two types of vector operations: Either translations or switches. The number of switches performed is a key factor of the complexity bounds. The H-LLL component of the cost of $\widetilde{\mathrm{L}}^1$ could be lowered by using faster LLL-reducing algorithms than H-LLL (with respect to d), but for our amortization to hold, they have to satisfy a standard property (see Section 3.2). The term $\mathcal{O}(d^{\omega+1+\varepsilon}\beta^{1+\varepsilon})$ derives from both the HNF computation mentioned above and a series of product trees of balanced matrix multiplications whose overall product has bit-length $\mathcal{O}(d\beta)$. Furthermore, the precise cost dependence of $\widetilde{\mathrm{L}}^1$ in β is $\mathcal{P}oly(d)\cdot\mathcal{M}(\beta)\log\beta$. We also remark that the cost can be proven to be $\mathcal{O}(d^{4+\varepsilon}\log|\det B| + d^{5+\varepsilon} + d^{\omega}(\log|\det B|)^{1+\varepsilon}) + \mathcal{H}(d,\beta)$, where $\mathcal{H}(d,\beta)$ denotes the cost of computing the Hermite normal form. Finally, we may note that if the size-reduction parameter θ is not considered as a constant, then a factor $\mathcal{P}oly(\log(1/\theta))$ is involved in the cost of the leaf calls.

Road-map. We construct $\widetilde{\mathrm{L}}^1$ in several generalization steps which, in the gcd framework, respectively correspond to Euclid's algorithm (Section 2), Lehmer's inclusion of truncations in Euclid's algorithm (Section 3) and the Knuth-Schönhage recursive generalization of Lehmer's algorithm (Section 4).

2 Lift-Reduction

In order to enable the adaptation of the gcd framework to lattice reduction, we introduce a new type of reduction which behaves more predictively and regularly. In this new framework, called lift-reduction, we are given a reduced matrix B and a lifting target $\ell \geq 0$, and we aim at computing a unimodular U such that $\sigma_\ell BU$ is reduced (with $\sigma_\ell = \mathrm{diag}(2^\ell, 1, \ldots, 1)$). Lift-reduction can naturally be performed using any general purpose reduction algorithm, however we will design fast algorithms specific to lift-reduction in Sections 3 and 4. Lifting a lattice basis has a predictable impact on the $r_{i,i}$'s and the successive minima.

Lemma 4. *Let B be non-singular and $\ell \geq 0$. If R (resp. R') is the R-factor of B (resp. $B' = \sigma_\ell B$), then $r'_{i,i} \geq r_{i,i}$ for all i and $\prod r'_{i,i} = 2^\ell \prod r_{i,i}$. Furthermore, if $(\lambda_i)_i$ (resp. $(\lambda'_i)_i$) are the successive minima of $L = L(B)$ (resp. $L' = L(B')$), then $\lambda_i \leq \lambda'_i \leq 2^\ell \lambda_i$ for all i.*

Proof. The first statement is proven in [10, Le. 4]. For the second one, notice that $\prod r'_{i,i} = |\det B'| = 2^\ell |\det B| = 2^\ell \prod r_{i,i}$. We now prove the third statement. Let $(\mathbf{v}_i)_i$ and $(\mathbf{v}'_i)_i$ be linearly independent vectors in L and L' respectively with $\|\mathbf{v}_i\| = \lambda_i$ and $\|\mathbf{v}'_i\| = \lambda'_i$ for all i. For any i, we define $S'_i = \{\sigma_\ell \mathbf{v}_j, j \leq i\}$ and $S_i = \{\sigma_\ell^{-1}\mathbf{v}'_j, j \leq i\}$. These are linearly independent sets in L' and L respectively. Then for any i we have $\lambda_i \leq \max_{\|\cdot\|}(S_i) \leq \lambda'_i \leq \max_{\|\cdot\|}(S'_i) \leq 2^\ell \lambda_i$. □

We can now bound the entries of any matrix which performs lift-reduction.

Lemma 5. *Let Ξ_1, Ξ_2 be valid parameters and α_1 and α_2 as in Theorem 1. Let $\ell \geq 0$, $B \in \mathbb{R}^{d\times d}$ be Ξ_1-reduced and U such that $C = \sigma_\ell BU$ is Ξ_2-reduced. Letting $\zeta_1 = (1+\eta_1+\theta_1)\alpha_1\alpha_2$, we have:*

$$\forall i,j:\ |u_{i,j}| \leq 4d^3\zeta_1^d \cdot \frac{r'_{j,j}}{r_{i,i}} \leq 2^{\ell+2}d^3\zeta_1^{2d} \cdot \frac{r_{j,j}}{r_{i,i}},$$

where R (resp. R') is the R-factor of B (resp. C). In addition, if $V = U^{-1}$ and $\zeta_2 = (1+\eta_2+\theta_2)\alpha_2\alpha_1$:

$$\forall i,j:\ |v_{j,i}| \leq 2^{\ell+2}d^3\zeta_2^d \cdot \frac{r_{i,i}}{r'_{j,j}} \leq 2^{\ell+2}d^3\zeta_2^{2d} \cdot \frac{r_{i,i}}{r_{j,j}}.$$

Proof. Let $B = QR$, $C = Q'R'$ be the QR-factorizations of B and C. Then

$$\begin{aligned} U &= R^{-1}Q^t\sigma_\ell^{-1}Q'R' \\ &= \mathrm{diag}(r_{i,i}^{-1})\bar{R}^{-1}\left(Q^t\sigma_\ell^{-1}Q'\right)\bar{R}'\mathrm{diag}(r'_{j,j}), \end{aligned}$$

with $\bar{R} = R \cdot \mathrm{diag}(1/r_{i,i})$ and $\bar{R}' = R' \cdot \mathrm{diag}(1/r'_{j,j})$. From the proof of [2, Le. 5.5], we know that $|\bar{R}^{-1}| \leq 2((1+\eta_1+\theta_1)\alpha_1)^d T$, where $t_{i,j} = 1$ if $i \leq j$ and $t_{i,j} = 0$ otherwise. By Theorem 1, we have $|\bar{R}'| \leq (\eta_2\alpha_2^{d-1}+\theta_2)T \leq 2\alpha_2^d T$ (using $\theta_2 \leq \alpha_2$ and $\eta_2 \leq 1$). Finally, we have $|Q|, |Q'| \leq M$, where $m_{i,j} = 1$ for all i,j. Using the triangular inequality, we obtain:

$$\begin{aligned} |U| &\leq 4\zeta^d\mathrm{diag}(r_{i,i}^{-1})TM^2T\mathrm{diag}(r'_{j,j}) \\ &\leq 4d^3\zeta^d\mathrm{diag}(r_{i,i}^{-1})M\mathrm{diag}(r'_{j,j}). \end{aligned}$$

Now, by Theorem 1 and Lemma 4, we have $r'_{j,j} \leq \alpha_2^{d-j}\lambda'_j \leq 2^\ell\alpha_2^{d-j}\lambda_j \leq 2^\ell\alpha_1^j\alpha_2^{d-j}r_{j,j}$, which completes the proof of the first statement.

For the second statement note that

$$V = \mathrm{diag}(r'^{-1}_{i,i})\bar{R}'^{-1}\left(Q'^t\sigma_\ell Q\right)\bar{R}\mathrm{diag}(r_{j,j})$$

is similar to the expression for U in the proof of the first statement, except that σ_ℓ can increase the innermost product by a factor 2^ℓ. □

LLL-reduction as a sequence of lift-reductions. In the remainder of this section we illustrate that LLL-reduction can be achieved with an efficient sequence of lift-reductions.

Lift-reduction is specialized to reducing a scalar-shift/lift of an already reduced basis. In Figure 2 we create reduced bases (of distinct lattices from the input lattice) which we use to progressively create a reduced basis for the input lattice. Here we use an HNF triangularization and scalar shifts to find suitable reduced lattice bases. We analyze the cost and accuracy of Figure 2 using a generic lift-reduction algorithm. The remainder of the paper can then focus on specialized lift-reduction algorithms which each use Figure 2 to achieve generic reduction. We note that other wrappers of lift-reduction are possible.

Recall that the HNF of a (full-rank) lattice $L \subseteq \mathbb{Z}^d$ is the unique upper triangular basis H of L such that $-h_{i,i}/2 \leq h_{i,j} < h_{i,i}/2$ for any $i < j$ and $h_{i,i} > 0$ for any i. Using [17, 31], it can be computed in time $O(d^{\omega+1+\varepsilon}\beta^{1+\varepsilon})$, where the input matrix $B \in \mathbb{Z}^{d\times d}$ satisfies $\max\|\mathbf{b}_i\| \leq 2^\beta$.

Let H be the HNF of $L(B)$. At the end of Step 1, the matrix $B = H$ is upper triangular, $\prod b_{i,i} = |\det H| \leq 2^{d\beta}$, and the 1×1 bottom rightmost sub-matrix of H is trivially Ξ-reduced.

In each iteration we Ξ-reduce a lower-right sub-matrix of B via lift-reduction (increasing the dimension with each iteration). This is done by augmenting the previous Ξ-reduced sub-matrix by a scaling down of the next row (such that the new values are tiny). This creates a C which is reduced and such that a lift-reduction of C will be a complete Ξ-reduction of the next largest sub-matrix of B. The column operations of the lift-reduction are then applied to rest of B with the triangular structure allowing us to reduce each remaining row modulo $b_{i,i}$. From a cost point of view, it is worth noting that the sum of the lifts ℓ_k is $\mathcal{O}(\log|\det H|) = \mathcal{O}(d\beta)$.

Inputs: LLL parameters Ξ; a non-singular $B \in \mathbb{Z}^{d\times d}$.
Output: A Ξ-reduced basis of $L(B)$.

1. $B := \mathrm{HNF}(B)$.
2. For k from $d-1$ down to 1 do
3. Let C be the bottom-right $(d-k+1)$-dimensional submatrix of B.
4. $\ell_k := \lceil \log_2(b_{k,k}) \rceil$, $C := \sigma_{\ell_k}^{-1} C$.
5. *Lift-reduction:* Find U' unimodular such that $\sigma_{\ell_k} C U'$ is Ξ-reduced.
6. Let U be the block-diagonal matrix $\mathrm{diag}(I, U')$.
7. Compute $B := B \cdot U$, reducing row i symmetrically modulo $b_{i,i}$ for $i < k$.
8. Return B.

Fig. 2. Reducing LLL-reduction to lift-reduction.

Lemma 6. *The algorithm of Figure 2 Ξ-reduces B such that* $\max\|\mathbf{b}_i\| \leq 2^\beta$ *using*

$$\mathcal{O}(d^{\omega+1+\varepsilon}(\beta^{1+\varepsilon}+d)) + \sum_{k=d-1}^{1} \mathcal{C}_k$$

bit operations, where $\mathcal{C}_k$ is the cost of Step 5 for the specific value of k.

Proof. We first prove the correctness of the algorithm. We let U_H be the unimodular transformation such that $H = BU_H$. For $k < d$, we let U'_k be the $(d-k+1)\times(d-k+1)$ unimodular transformation that reduces $\sigma_{\ell_k} C$ at Step 5 and U''_k be the unimodular transformation that reduces rows $1 \leq i < k$ at Step 7. With input B the algorithm returns $B \cdot U_H \cdot \mathrm{diag}(I, U'_{d-1}) \cdot U''_{d-1} \ldots \cdot \mathrm{diag}(I, U'_2) \cdot U''_2 \cdot U'_1$. Since B is multiplied by a product of unimodular matrices, the output matrix is a basis of the lattice spanned by the columns of B.

We show by induction on k from d down to 1 that at the end of the $(d-k)$-th loop iteration, the bottom-right $(d-k+1)$-dimensional submatrix of the current B is Ξ-reduced. The statement is valid for $k = d$, as a non-zero matrix in dimension 1 is always reduced, and instanciating the statement with $k = 1$ ensures that the matrix resturned by the algorithm is Ξ-reduced. The non-trivial ingredient of the proof of the statement is to show that for $k < d$, the input of the *lift-reduction* of Step 5 is valid, i.e., that at the beginning of Step 5 the matrix C is Ξ-reduced. Let R be the R-factor of C. Let C' be the bottom-right $(d-k)\times(d-k)$ submatrix of C. By induction, we know that C' is Ξ-reduced. It thus remains to show that the first row of R satisfies the size-reducedness condition, and that Lovász' condition between the first two rows is satisfied. We have $r_{1,j} = h_{k,k+j-1}/2^{\ell_k}$, for $j \leq d-k+1$, thus ensuring the size-reducedness condition. Furthermore, by the shape of the unimodular transformations applied so far, we know that C' is a basis of the lattice L' generated by the columns of the bottom-right $(d-k)$-dimensional submatrix of H, which has first minimum $\lambda_1(L') \geq \min_{i>k} h_{i,i} \geq 1$. As $r_{2,2}$ is the norm of the first vector of C', we have $r_{2,2} \geq \lambda_1(L') \geq 1$. Independently, by choice of ℓ_k, we have $r_{1,1} \leq 1$. This ensures that Lovász' condition is satisfied, and completes the proof of correctness.

We now bound the cost of the algorithm of Figure 2. We bound the overall cost of the $d-1$ calls to lift-reduction by $\sum_{k<d} \mathcal{C}_k$. It remains to bound the contribution of Step 7 to the cost. The cost dominating component of Step 7 is the computation of the product of the last $d-k+1$ columns of (the current value of) B by U'. We consider separately the costs of computing the products by U' of the $k \times (d-k+1)$ top-right submatrix $\overline{B}$ of B, and of the $(d-k) \times (d-k+1)$ bottom-right submatrix $\underline{B}$ of B

For $i \leq k$, the magnitudes of the entries of the i-th row of $\overline{B}$ are uniformly bounded by $h_{i,i}$. By Lemma 5, if $e, j < d-k+1$, then $|u'_{e,j}| \leq 2^{\ell_k+2} d^3 \zeta_1^d \cdot \frac{r_{j,j}}{r_{e,e}}$ (recall that R is the R-factor of C at the beginning of Step 5). As we saw above, we have $r_{2,2} \geq 1$, and, by reducedness, we have $r_{e,e} \geq \alpha^{-e}$ for any $e \geq 2$ (using Theorem 1). Also, by choice of ℓ_k, we have $r_{1,1} \geq 1/2$. Overall, this gives that the jth column of U' is uniformly bounded as $\log \|\mathbf{u}'_j\| = \mathcal{O}(\ell_k + d + \log r_{j,j})$. The bounds on the bit-lengths of the rows of $\overline{B}$ and the bounds on the bit-lengths of the columns of U' may be very unbalanced. We do not perform matrix multiplication naively, as this unbalancedness may lead to too large a cost (the maxima of row and column bounds may be much larger than the averages). To circumvent this difficulty we use Recipe 1, given in Appendix 1 p. 17, with "$S = \log\det H + d^2 + d\ell_k$". Since $\det H = |\det B|$ the multiplication of $\overline{B}$ with U' can be performed within $\mathcal{O}(d^\omega \mathcal{M}((\log|\det B|)/d + d + \ell_k))$ bit operations.

We now consider the product $P := \underline{B}U'$. By reducedness of $\underline{B}$, we have $\|\underline{\mathbf{b}}_j\| \leq \alpha^d r_{j,j}$ (from Theorem 1). Recall that we have $|u'_{e,j}| \leq 2^{\ell_k+2} d^3 \zeta_1^d \cdot \frac{r_{j,j}}{r_{e,e}}$. As a consequence, we can uniformly bound $\log\|\mathbf{u}'_j\|$ and $\log\|\mathbf{p}_j\|$ by $\mathcal{O}(\ell_k + d + \log r_{j,j})$ for any j. We can thus use Recipe 3, given in Appendix 1 p. 17, to compute P, with "$S = \mathcal{O}(\log\det H + d^2 + d\ell_k)$" using $\mathcal{O}(d^{\omega+\varepsilon}\mathcal{M}((\log|\det B|)/d + d + \ell_k))$ bit operations.

The proof can be completed by noting that the above matrix products are performed $d-1$ times during the execution of the algorithm and by also considering the cost $\mathcal{O}(d^{\omega+1+\varepsilon}\beta^{1+\varepsilon})$ of converting B to Hermite normal form. □

We use the term $\mathcal{C}_k$ in order to amortize over the loop iterations the costs of the calls to the lift-reducing algorithm. In the algorithm of Figure 2 and in Lemma 6, the lift-reducing algorithm is not specified. It may be a general-purpose LLL-reducing algorithm [16, 11, 21, 20] or a specifically designed lift-reducing algorithm such as Lift-$\widetilde{\mathrm{L}}^1$, described in Section 4.

It can be noted from the proof of Lemma 6 that the non-reduction costs can be refined as $\mathcal{O}(d^{\omega+\varepsilon}\mathcal{M}(\log|\det B|) + d^{\omega+1+\varepsilon}\mathcal{M}(d)) + \mathcal{H}(d,\beta)$. We note that the HNF is only used as a triangularization, thus any triangularization of the input B will suffice, however then it may be needed to perform d^2 reductions of entries $b_{i,j}$ modulo $b_{i,i}$. Thus we could replace $\mathcal{H}(d,\beta)$ by $\mathcal{O}(d^2\beta^{1+\varepsilon})$ for upper triangular inputs. Using the cost of H-LLL for lift-reduction, we can bound the complexity of Figure 2 by $\mathcal{P}oly(d) \cdot \beta^2$. This is comparable to L^2 and H-LLL.

3 Truncating matrix entries

We will now focus on improving the lift-reduction step introduced in the previous section. In this section we show how to truncate the "remainder" matrix and we give an efficient factorization for the "quotient" matrices encountered in the process. This way the unimodular transformations can be found and stored at low cost. In the first part of this section, we show that given any B reduced and $\ell \geq 0$, finding U such that $\sigma_\ell BU$ is reduced can be done by looking at only the most significant bits of each column of B. In the context of gcd algorithms, this is equivalent to saying that the quotients can be computed by looking at the most significant bits of the remainders only. In the gcd case, using only the most significant bits of the remainders allows one to efficiently

compute the quotients. Unfortunately, this is where the gcd analogy stops as a lift-reduction transformation U may still have entries that are much larger than the number of bits kept of B. In particular, if the diagonal coefficients of the R-factor of B are very unbalanced, then Lemma 5 does not prevent some entries of U from being as large as the magnitudes of the entries of B (as opposed to just the precision kept). The second part of this section is devoted to showing how to make the bit-size of U and the cost of computing it essentially independent of these magnitudes. In this framework we can then describe and analyze a Lehmer-like lift-reduction algorithm.

3.1 The most significant bits of B suffice for reducing $\sigma_\ell B$

It is a natural strategy to reduce a truncation of B rather than B, but in general it is unclear if some U which reduces a truncation of B would also reduce B even in a weaker sense. However, with lift-reduction we can control the size of U which allows us to overcome this problem. In this section we aim at computing a unimodular U such that $\sigma_\ell BU$ is reduced, when B is reduced, by working on a truncation of B. We use the bounds of Lemma 5 on the magnitude of U to show that a column-wise truncation precision of $\ell + \mathcal{O}(d)$ bits suffices for that purpose.

Lemma 7. *Let Ξ_1, Ξ_2, Ξ_3 be valid reduction parameters with $\Xi_3 > \Xi_2$. There exists a constant c_3 such that the following holds for any $\ell \geq 0$. Let $B \in \mathbb{R}^{d\times d}$ be Ξ_1-reduced and ΔB be such that $\max \frac{\|\Delta \mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-\ell - c_3 \cdot d}$. If $\sigma_\ell(B + \Delta B)U$ is Ξ_3-reduced for some U, then $\sigma_\ell BU$ is Ξ_2-reduced.*

The proof is given in Appendix 2 p. 19. The above result implies that to find a U such that $\sigma_\ell BU$ is reduced, it suffices to find U such that $\sigma_\ell(B' \cdot E)U$ is reduced (for a stronger Ξ), for well chosen matrices B' and E, outlined as follows.

Definition 2. *For $B \in \mathbb{Z}^{d\times d}$ with $\beta = \log \max \|\mathbf{b}_j\|$ and precision p, we chose to store the p most significant bits of B, $\mathrm{MSB}_p(B)$, as a matrix product $B'E$ or just the pair (B', E). This pair should satisfy $B' \in \mathbb{Z}^{d\times d}$ with $p = \log \max \|\mathbf{b'}_j\|$, $E = \mathrm{diag}(2^{e_i - p})$ with $e_i \in \mathbb{Z}$ such that $\frac{2^{e_i} - \|\mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^d$, and $\max \frac{\|(\mathbf{b}_j - \mathbf{b'}_j \cdot 2^{e_i - p}\|}{\|\mathbf{b}_j\|} \leq 2^{-p}$.*

3.2 Finding a unimodular U reducing $\sigma_\ell B$ at low cost

The algorithm `TrLiftLLL` (a truncated lift-LLL) we propose is an adaptation of the `StrengthenLLL` from [19], which aims at strengthening the LLL-reducedness of an already reduced basis, i.e., Ξ_2-reducing a Ξ_1-reduced basis with $\Xi_1 < \Xi_2$. One can recover a variant of `StrengthenLLL` by setting $\ell = 0$ below. We refer the reader to Appendix 3 p. 19 for a complete description of `TrLiftLLL`.

Theorem 2. *For any valid parameters $\Xi_1 < \Xi_2$ and constant c_4, there exists a constant c_4' and an algorithm `TrLiftLLL` with the following specifications. It takes as inputs $\ell \geq 0$, $B \in \mathbb{Z}^{d\times d}$ and $E = \mathrm{diag}(2^{e_i})$ with $\max \|\mathbf{b}_i\| \leq 2^{c_4(\ell + d)}$, $e_i \in \mathbb{Z}$ and BE is Ξ_1-reduced; It runs in time $O(d^{2+\varepsilon}(d+\ell)(d+\ell+\tau) + d^2 \log \max(1 + |e_i|))$, where $\tau = O(d^2(\ell + d))$ is the number of switches performed during the single call it makes to H-LLL; And it returns two matrices U and D such that:*

1. *$D = \mathrm{diag}(2^{d_i})$ with $d_i \in \mathbb{Z}$ satisfying $\max |e_i - d_i| \leq c_4'(\ell + d)$,*
2. *U is unimodular and $\max |u_{i,j}| \leq 2^{\ell + c_4' \cdot d}$,*
3. *$D^{-1}UD$ is unimodular and $\sigma_\ell(BE)(D^{-1}UD)$ is Ξ_2-reduced.*

When setting $\ell = \mathcal{O}(d)$, we obtain the base case of lift-$\widetilde{\mathrm{L}}^1$, the quasi-linear time recursive algorithm to be introduced in the next section. The most expensive step of `TrLiftLLL` is a call to an LLL-type algorithm, which must satisfy a standard property that we identify hereafter.

When called on a basis matrix B with R-factor R, the L^3, L^2 and H-LLL algorithms perform two types of basis operations: They either subtract to a vector $\mathbf{b}_k$ an integer combination of $\mathbf{b}_1, \ldots, \mathbf{b}_{k-1}$ (translation), or they exchange $\mathbf{b}_{k-1}$ and $\mathbf{b}_k$ (switches). Translations leave the $r_{i,i}$'s unchanged. Switches are never perfomed when the optimal Lovász condition $r_{i,i}^2 \leq r_{i,i+1}^2 + r_{i+1,i+1}^2$ is satisfied, and thus cannot increase any of the quantities $\max_{j \leq i} r_{j,j}$ (for varying i), nor decrease any of the quantities $\min_{j \geq i} r_{j,j}$. This implies that if we have $\max_{i<k} r_{i,i} < \min_{i \geq k} r_{i,i}$ for some k at the beginning of the execution, then the computed matrix U will be such that $u_{i,j} = 0$ for any (i,j) such that $i \geq k$ and $j < k$. We say that a LLL-reducing algorithm satisfies Property (P) if for any k such that $\max_{i<k} r_{i,i} < \min_{i \geq k} r_{i,i}$ holds at the beginning of the execution, then it also holds at the end of the execution.

Property (P) is for instance satisfied by L^3 ([16, p. 523]), L^2 ([21, Th. 6]) and H-LLL ([20, Th. 4.3]). We choose H-LLL as this currently provides the best complexity bound, although $\widetilde{\mathrm{L}}^1$ would remain quasi-linear with L^3 or L^2.

`TrLiftLLL` will also be used with $\ell = 0$ in the recursive algorithm for strengthening the reduction parameters. Such refinement is needed after the truncation of bases and transformation matrices which we will need to ensure that the recursive calls get valid inputs.

3.3 A Lehmer-like lift-LLL algorithm

By combining Lemma 7 and Theorem 2, we obtain a Lehmer-like Lift-LLL algorithm, given in Figure 3. In the input, we assume the base-case lifting target t divides ℓ. If it is not the case, we may replace ℓ by $t\lfloor \ell/t \rfloor$, and add some more lifting at the end.

Inputs: LLL parameters Ξ; a Ξ-reduced matrix $B \in \mathbb{Z}^{d \times d}$; a lifting target ℓ; a divisor t of ℓ.
Output: A Ξ-reduced basis of $\sigma_\ell B$.

1. Let Ξ_0, Ξ_1 be valid parameters with $\Xi_0 < \Xi < \Xi_1$,
   c_3 as in Le. 7 for "$(\Xi_1, \Xi_2, \Xi_3) := (\Xi, \Xi, \Xi_1)$",
   c_1 as in Le. 3 with "$(\Xi_1, \Xi_2) := (\Xi, \Xi_0)$",
   and c_4' as in Th. 2 with "$(\Xi_1, \Xi_2, c_4) := (\Xi_0, \Xi_1, c_3 + 2)$".
2. For k from 1 to ℓ/t do
3. $\quad (B', E) := \mathrm{MSB}_{(t+c_3 d)}(B)$.
4. $\quad (D, U) :=$ `TrLiftLLL`(B', E, t).
5. $\quad B := \sigma_t B D^{-1} U D$.
6. Return B.

Fig. 3. The `Lehmer-LiftLLL` algorithm.

Theorem 3. `Lehmer-LiftLLL` *is correct. Furthermore, if the input matrix* B *satisfies* $\max \|\mathbf{b}_i\| \leq 2^\beta$*, then its bit-complexity is* $O(d^3 \ell (d^{1+\varepsilon} t + t^{-1+\varepsilon}(\ell + \beta)))$.

Proof. The correctness is provided by Lemmata 3 and 7 and by Theorem 2. At any moment throughout the execution, the matrix B is a Ξ-reduced basis of the lattice spanned by an ℓ'-lift of the input, for some $\ell' \leq \ell$. Therefore, by Theorem 1 and Lemma 4, the inequality $\max \|\mathbf{b}_i\| \leq \alpha^d \max r_{i,i} \leq 2^{c \cdot (\ell + \beta)}$ holds throughout the execution, for some constant c. The cost of Step 3 is $O[d^2(t + \log(\ell + \beta))]$. The cost of Step 4 is $O[d^{4+\varepsilon} t^2 + d^2 \log(\ell + \beta)]$. Step 5 is performed by first

computing $\sigma_t BD^{-1}$, whose entries have bit-sizes $\mathcal{O}(\ell+\beta)$, and then multiplying by U and finally by D. This costs $\mathcal{O}(d^3(\ell+\beta)t^\varepsilon)$ bit operations. The claimed complexity bound can by obtained by summing over the ℓ/t loop iterations. □

Note that if ℓ is sufficiently large with respect to d, then we may choose $t = \ell^a$ for $a \in (0,1)$, to get a complexity bound that is subquadratic with respect to ℓ. By using `Lehmer-LiftLLL` at Step 5 of the algorithm of Figure 2 (with $t = \ell^{.5}$), it is possible to obtain an LLL-reducing algorithm of complexity $\mathcal{P}oly(d) \cdot \beta^{1.5+\varepsilon}$.

4 Quasi-linear algorithm

We now aim at constructing a recursive variant of the `Lehmer-LiftLLL` algorithm of the previous section. Because the lift-reducing unimodular transformations will be produced by recursive calls, we have little control over their structure (as opposed to those produced by `TrLiftLLL`). Before describing Lift-$\widetilde{\mathrm{L}}^1$, we thus study lift-reducing unimodular transformations, without considering how they were computed. In particular, we are interested in how to work on them at low cost. This study is robust and fully general, and afterwards is used to analyze lift-$\widetilde{\mathrm{L}}^1$.

4.1 Sanitizing unimodular transforms

In the previous section we have seen that working on the most significant bits of the input matrix B suffices to find a matrix U such that $\sigma_\ell BU$ is reduced. Furthermore, as shown in Theorem 2, the unimodular U can be found and stored on few bits. Since the complexity of Theorem 2 is quadratic in ℓ we will use it only for small lift-reductions (the leaves of our recursive tree) and repairing reduction quality (when $\ell = 0$). For large lifts we will use recursive lift-reduction. However, that means we no longer have a direct application of a well-understood LLL-reducing algorithm which was what allowed such efficient unimodular transforms to be found. Thus, in this section we show how any U which reduces $\sigma_\ell B$ can be transformed into a factored unimodular U' which also reduces $\sigma_\ell B$ and for which each entry can be stored with only $\mathcal{O}(\ell+d)$ bits. We also explain how to quickly compute the products of such factored matrices. This analysis can be used as a general framework for studying lift-reductions.

The following lemmata work because lift-reducing transforms have a special structure which we gave in Lemma 5. Here we show a class of additive perturbations which, when viewed as a transformations, are in fact unimodular transformations themselves. Note that these entry-wise perturbations are stronger than mere truncations since $\Delta u_{i,j}$ could be larger than $u_{i,j}$. Lemma 8 shows that a sufficiently small perturbation of a unimodular lift-reducing matrix remains unimodular.

Lemma 8. *Let Ξ_1, Ξ_2 be valid LLL parameters. There exists a contant c_7 such that the following holds for any $\ell \geq 0$. Let $B \in \mathbb{R}^{d\times d}$ (with R-factor R) be Ξ_1-reduced, and U be unimodular such that $\sigma_\ell BU$ (with R-factor R') is Ξ_2-reduced. If $\Delta U \in \mathbb{Z}^{d\times d}$ satisfies $|\Delta u_{i,j}| \leq 2^{-(\ell+c_7\cdot d)} \cdot \frac{r'_{j,j}}{r_{i,i}}$ for all i, j, then $U + \Delta U$ is unimodular.*

Proof. Since U is unimodular, the matrix $V = U^{-1}$ exists and has integer entries. We can thus write $U + \Delta U = U(I + U^{-1}\Delta U)$, and prove the result by showing that $U^{-1}\Delta U$ is strictly upper triangular, i.e., that $(U^{-1}\Delta U)_{i,j} = 0$ for $i \geq j$. We have $(U^{-1}\Delta U)_{i,j} = \sum_{k\leq d} v_{i,k} \cdot \Delta u_{k,j}$. We now show that if $\Delta u_{k,j} \neq 0$ and $i \geq j$, then we must have $v_{i,k} = 0$ (for a large enough c_7).

The inequality $\Delta u_{k,j} \neq 0$ and the hypothesis on ΔU imply that $\frac{r_{k,k}}{r'_{j,j}} \leq 2^{-(\ell+c_7\cdot d)}$. Since $i \geq j$ and $\sigma_\ell BU$ is reduced, Theorem 1 implies that $\frac{r_{k,k}}{r'_{i,i}} \leq 2^{-\ell+(c-c_7)d}$, for some constant $c > 0$.

By using the second part of Lemma 5, we obtain that there exists $c' > 0$ such that $|v_{i,k}| \leq 2^{\ell+c'\cdot d} \cdot \frac{r_{k,k}}{r'_{i,i}} \leq 2^{(c+c'-c_7)d}$. As V is integral, setting $c_7 > c + c'$ allows us to ensure that $v_{i,k} = 0$, as desired. □

Lemma 9 shows that a sufficiently small perturbation of a unimodular lift-reducing matrix remains lift-reducing.

Lemma 9. *Let Ξ_1, Ξ_2, Ξ_3 be valid LLL parameters such that $\Xi_2 > \Xi_3$. There exists a contant c_8 such that the following holds for any $\ell \geq 0$. Let $B \in \mathbb{R}^{d\times d}$ (with R-factor R) be Ξ_1-reduced, and U be unimodular such that $\sigma_\ell BU$ (with R-factor R') is Ξ_2-reduced. If $\Delta U \in \mathbb{Z}^{d\times d}$ satisfies $|\Delta u_{i,j}| \leq 2^{-(\ell+c_8\cdot d)} \cdot \frac{r'_{j,j}}{r_{i,i}}$ for all i, j, then $\sigma_\ell B(U + \Delta U)$ is Ξ_3-reduced.*

Proof. We proceed by showing that $|\sigma_\ell B\Delta U|$ is column-wise small compared to $|\sigma_\ell BU|$ and by applying Lemma 3. We have $|\Delta U| \leq 2^{-(\ell+c_8\cdot d)}\mathrm{diag}(r_{i,i}^{-1})C\mathrm{diag}(r'_{j,j})$ by assumption, where $c_{i,j} = 1$ for all i, j. Since B is Ξ_1-reduced, we also have $|R| \leq \mathrm{diag}(r_{i,i})T + \theta_1 T\mathrm{diag}(r_{j,j})$, where T is upper triangular with $t_{i,j} = 1$ for all $i \leq j$. Then using $|R\Delta U| \leq |R||\Delta U|$ we get

$$|R\Delta U| \leq 2^{-(\ell+c_8\cdot d)}\Big(\mathrm{diag}(r_{i,i})T\mathrm{diag}(r_{j,j}^{-1}) + \theta_1 T\Big)C\mathrm{diag}(r'_{j,j}).$$

Since B is Ξ_1-reduced, by Theorem 1, we have $r_{i,i} \leq \alpha_1^d r_{j,j}$ for all $i \leq j$, hence it follows that

$$|R\Delta U| \leq 2^{-(\ell+c_8\cdot d)}(\alpha_1^d + \theta_1)TC\mathrm{diag}(r'_{j,j}).$$

As a consequence, there exists a constant $c > 0$ such that for any j:

$$\|(\sigma_\ell B\Delta U)_j\| \leq 2^\ell \|(B\Delta U)_j\| = 2^\ell \|(R\Delta U)_j\| \leq 2^{(c-c_8)d} r'_{j,j}.$$

We complete the proof by noting that $r'_{j,j} \leq \|(\sigma_\ell BU)_j\|$ and by applying Lemma 3 (which requires that c_8 is set sufficiently large). □

Lemmata 8 and 9 allow us to design an algorithmically efficient representation for lift-reducing unimodular transforms.

Theorem 4. *Let Ξ_1, Ξ_2, Ξ_3 be valid LLL parameters with $\Xi_2 > \Xi_3$. There exist contants $c_9, c_{10} > 0$ such that the following holds for any $\ell \geq 0$. Let $B \in \mathbb{R}^{d\times d}$ be Ξ_1-reduced, and U be unimodular such that $\sigma_\ell BU$ is Ξ_2-reduced. Let $d_i := \lfloor \log \|\mathbf{b}_i\| \rfloor$ for all i. Let $D := \mathrm{diag}(2^{d_i})$, $x := \ell + c_9 \cdot d$, $\widehat{U} := 2^x DUD^{-1}$ and $U' := 2^{-x}D^{-1}\lfloor \widehat{U} \rfloor D$. We write $\mathtt{Clean}(U, (d_i)_i, \ell) := (U', D, x)$. Then U' is unimodular and $\sigma_\ell BU'$ is Ξ_3-reduced. Furthermore, the matrix $\widehat{U}$ satisfies $\max|\widehat{u}_{i,j}| \leq 2^{2\ell+c_{10}\cdot d}$.*

Proof. We first show that U' is integral. If $\lfloor \widehat{u}_{i,j} \rfloor = \widehat{u}_{i,j}$, then $u'_{i,j} = u_{i,j} \in \mathbb{Z}$. Otherwise, we have $\widehat{u}_{i,j} \notin \mathbb{Z}$, and thus $x + d_i - d_j \leq 0$. This gives that $\lfloor \widehat{u}_{i,j} \rfloor \in \mathbb{Z} \subseteq 2^{x+d_i-d_j}\mathbb{Z}$. We conclude that $u'_{i,j} \in \mathbb{Z}$.

Now, consider $\Delta U = U' - U$. Since $\Delta U = 2^{-x}D^{-1}(\lfloor \widehat{U} \rfloor - \widehat{U})D$, we have $|\Delta u_{i,j}| \leq 2^{d_j-d_i-x}$, for all i, j. Thus by Theorem 1 and Lemma 4, we have $|\Delta u_{i,j}| \leq 2^{-x+c\cdot d} \cdot \frac{r'_{j,j}}{r_{i,i}}$ for some constant c. Applying Lemmata 8 and 9 shows that U' is unimodular and $\sigma_\ell BU'$ is Ξ_3-reduced (if c_9 is chosen sufficiently large).

By Lemma 5, we have for all i, j:

$$|\widehat{u}_{i,j}| = |u_{i,j}|2^{x+d_i-d_j} \leq 2^{x+\ell+c'd} \cdot \frac{r_{j,j}}{2^{\lfloor \log \|\mathbf{b}_j\| \rfloor}} \frac{2^{\lfloor \log \|\mathbf{b}_i\| \rfloor}}{r_{i,i}},$$

for some constant c'. Theorem 1 then provides the result. □

The above representation of lift-reducing transforms is computationally powerful. Firstly, it can be efficiently combined with Theorem 2: Applying the process described in Theorem 4 to the unimodular matrix produced by `TrLiftLLL` may be performed in $O(d^2(d+\ell)+d\log\max(1+|e_i|))$ bit operations, which is negligible comparable to the cost bound of `TrLiftLLL`. We call `TrLiftLLL'` the algorithm resulting from the combination of Theorems 2 and 4. `TrLiftLLL'` is to be used as base case of the recursion process of `Lift-`$\widetilde{\mathrm{L}}^1$. Secondly, the following result shows how to combine lift-LLL-reducing unimodular transforms. This is an engine of the recursion process of `Lift-`$\widetilde{\mathrm{L}}^1$.

Lemma 10. *Let $U = 2^{-x}D^{-1}U'D \in \mathbb{Z}^{d\times d}$ with $U' \in \mathbb{Z}^{d\times d}$ and $D = \mathrm{diag}(2^{d_i})$. Let $V = 2^{-y}E^{-1}V'E \in \mathbb{Z}^{d\times d}$ with $V' \in \mathbb{Z}^{d\times d}$ and $E = \mathrm{diag}(2^{e_i})$. Let $\ell \in \mathbb{Z}$ and $f_i \in \mathbb{Z}$ for $i \leq d$. Then it is possible to compute the output (W', F, z) of* `Clean`$(U \cdot V, (f_i)_i, \ell)$ *(see Theorem 4) from x, y, ℓ, U', V', $(d_i)_i$, $(e_i)_i$, $(f_i)_i$, in time $O(d^\omega\mathcal{M}(t+\log d))$, where*

$$\max_{i,j}\max(|u'_{i,j}|, |v'_{i,j}|) \leq 2^t$$

and

$$\max_{i}\max(|d_i - e_i|, |f_i - e_i|, |\ell - (x+y)|) \leq t.$$

For short, we will write $W := U \odot V$, with $W = 2^{-z}F^{-1}W'F$ and $F = \mathrm{diag}(2^{f_i})$.

Proof. We first compute $m = \max|d_i - e_i|$. We have

$$UV = 2^{(-x-y-m)} \cdot F^{-1}T \cdot F,$$

where

$$T = (FD^{-1})U'\mathrm{diag}(2^{d_i-e_i+m})V'(EF^{-1}).$$

Then we compute T. We multiply U' by $\mathrm{diag}(2^{d_i-e_i+m})$, which is a mere multiplication by a non-negative power of 2 of each column of U'. This gives an integral matrix with coefficients of bit-sizes $\leq 3t$. We then multiply the latter by V', which costs $\mathcal{O}(d^\omega\mathcal{M}(t+\log d))$. We multiply the result from the left by (FD^{-1}) and from the right by EF^{-1}. From T, the matrix $\widehat{W}$ of Theorem 4 may be computed and rounded within $\mathcal{O}(d^2t)$ bit operations. □

It is crucial in the complexity analysis of `Lift-`$\widetilde{\mathrm{L}}^1$ that the cost of the merging process above is independent of the magnitude scalings (d_i, e_i and f_i).

4.2 `Lift-`$\widetilde{\mathrm{L}}^1$ algorithm

The `Lift-`$\widetilde{\mathrm{L}}^1$ algorithm given in Figure 4 relies on two recursive calls, on MSB, truncations, and on calls to `TrLiftLLL'`. The latter is used as base case of the recursion, and also to strengthen the reducedness parameters (to ensure that the recursive calls get valid inputs). When strengthening, the lifting target is always 0, and we do not specify it explicitly in Figure 4.

Theorem 5. `Lift-`$\widetilde{\mathrm{L}}^1$ *is correct.*

Proof. When $\ell \leq d$ the output is correct by Theorems 2 and 4. In Step 2, Theorems 2 and 4 give that BU_1 is Ξ_2-reduced and that U_1 has the desired format. In Step 3, the constant $c_3 \geq c_1$ is chosen so that Lemma 3 applies now and Lemma 7 will apply later in the proof. Thus B_1 is Ξ_1-reduced and has the correct structure by definition of MSB. Step 4 works (by induction) because B_1 satisfies the input requirements of `Lift-`$\widetilde{\mathrm{L}}^1$. Thus $\sigma_{\ell/2}B_1U_{R_1}$ is Ξ_1-reduced. Because of the selection of c_3 in Step 3 we know also that $\sigma_{\ell/2}BU_1U_{R_1}$ is reduced (weaker than Ξ_1) using

Lemma 7. Thus by Theorem 4, the matrix B_2 is reduced (weakly) and has an appropriate format for `TrLiftLLL'`. By Theorem 2, the matrix $\sigma_{\ell/2}BU_{1R_1}U_2$ is Ξ_3-reduced and by Theorem 4 we have that $\sigma_{\ell/2}BU_{1R_12}$ is Ξ_2-reduced. By choice of c_3 and Lemma 3, we know that the matrix B_3 is Ξ_1-reduced and satisfies the input requirements of `Lift-`$\widetilde{\mathrm{L}}^1$. Thus, by recursion, we know that $\sigma_{\ell/2}B_3U_{R_2}$ is Ξ_1-reduced. By choice of c_3 and Lemma 7, the matrix $\sigma_\ell BU_{1R_12}U_{R_2}$ is weakly reduced. By Theorem 4, the matrix B_4 is reduced and satisfies the input requirements of `TrLiftLLL'`. Therefore, the matrix $\sigma_\ell BU_{1R_12R_2}$ is Ξ_4-reduced. Theorem 4 can be used to ensure U has the correct format and $\sigma_\ell BU$ is Ξ_1-reduced. □

Inputs: Valid LLL-parameters $\Xi_3 > \Xi_2 \geq \Xi_4 > \Xi_1$; a lifting target ℓ;
$(B', (e_i)_i)$ such that $B = B'\mathrm{diag}(2^{e_i})$ is Ξ_1-reduced and $\max|b'_{i,j}| \leq 2^{\ell+c\cdot d}$.

Output: $(U', (d_i)_i, x)$ such that $\sigma_\ell BU$ is Ξ_1-reduced,
with $U = 2^{-x}\mathrm{diag}(2^{-d_i})U'\mathrm{diag}(2^{d_i})$ and $\max|u'_{i,j}| \leq 2^{2\ell+2c\cdot d}$.

1. If $\ell \leq d$, then use `TrLiftLLL'` with lifting target ℓ.
   Otherwise:
2. Call `TrLiftLLL'` on (B, Ξ_2); Let U_1 be the output. /* *Prepare 1st recursive call* */
3. $B_1 := \mathrm{MSB}_{(\ell/2+c_3\cdot d)}(B \cdot U_1)$.
4. Call `Lift-L1` on B_1, with lifting target $\ell/2$; /* *1st recursive call* */
   Let U_{R_1} be the output.
5. $U_{1R_1} := U_1 \odot U_{R_1}$. /* *Prepare 2nd recursive call* */
6. $B_2 := \sigma_{\ell/2}BU_{1R_1}$.
7. Call `TrLiftLLL'` on (B_2, Ξ_3). Let U_2 be the output.
8. $U_{1R_12} := U_{1R_1} \odot U_2$.
9. $B_3 := \mathrm{MSB}_{(\ell/2+c_3\cdot d)}(\sigma_{\ell/2}BU_{1R_12})$.
10. Call `Lift-L1` on B_3, with lifting target $\ell/2$; /* *2nd recursive call* */
    Let U_{R_2} be the output.
11. $U_{1R_12R_2} := U_{1R_12} \odot U_{R_2}$. /* *Prepare output* */
12. $B_4 := \sigma_\ell BU_{1R_12R_2}$.
13. Call `TrLiftLLL'` on (B_4, Ξ_4); Let U_3 be the output.
14. $U := U_{1R_12R_2} \odot U_3$; Return U.

Fig. 4. The `Lift-`$\widetilde{\mathrm{L}}^1$ algorithm.

4.3 Complexity analysis

Theorem 6. `Lift-`$\widetilde{\mathrm{L}}^1$ *has bit-complexity*

$$\mathcal{O}\left(d^{3+\varepsilon}(d+\ell+\tau) + d^\omega \mathcal{M}(\ell)\log\ell + \ell\log(\beta+\ell)\right),$$

where τ is the total number of LLL-switches performed by the calls to H-LLL (through `TrLiftLLL`*), and* $\max|b_{i,j}| \leq 2^\beta$.

Proof. We first bound the total cost of the calls to `TrLiftLLL'`. There are $\mathcal{O}(1+\ell/d)$ such calls, and for any of these the lifting target is $\mathcal{O}(d)$. Their contribution to the cost of `Lift-`$\widetilde{\mathrm{L}}^1$ is therefore $\mathcal{O}(d^{3+\varepsilon}(d+\ell+\tau))$. Also, the cost of handling the exponents in the diverse diagonal matrices is $\mathcal{O}(d(1+\ell/d)\log(\beta+\ell))$.

Now, let $\mathcal{C}(d,\ell)$ be the cost of the remaining operations performed by `Lift-`$\widetilde{\mathrm{L}}^1$, in dimension d and with lifting target ℓ. If $\ell \leq d$, then $\mathcal{C}(d,\ell) = \mathcal{O}(1)$ (as the cost of `TrLiftLLL'` has been put aside). Assume now that $\ell > d$. The operations to be taken into account include two recursive

calls (each of them costing $\mathcal{C}(d, \ell/2)$), and $\mathcal{O}(1)$ multiplications of d-dimensional integer matrices whose coefficients have bit-length $\mathcal{O}(d+\ell)$. This leads to the inequality $\mathcal{C}(d,\ell) \leq 2\mathcal{C}(d,\ell/2) + K \cdot d^\omega \mathcal{M}(d+\ell)$, for some absolute constant K. This leads to $\mathcal{C}(d,\ell) = \mathcal{O}(d^\omega \mathcal{M}(d+\ell)\log(d+\ell))$. □

4.4 $\widetilde{\mathbf{L}}^{\mathbf{1}}$ algorithm

The algorithm of Figure 4 is the Knuth-Schönhage-like generalization of the Lehmer-like algorithm of Figure 3. Now we are ready to analyze a general lattice reduction algorithm by creating a wrapper for Lift-$\widetilde{\mathrm{L}}^1$.

Algorithm $\widetilde{\mathrm{L}}^1$: We define $\widetilde{\mathrm{L}}^1$ as the algorithm from Figure 2, where Figure 5 is used to implement lift-reduction.

As we will see Figure 5 uses the truncation process MSB described in Definition 2 and `TrLiftLLL` to ensure that $\widetilde{\mathrm{L}}^1$ provides valid inputs to `Lift-`$\widetilde{\mathrm{L}}^1$. Its function is to process the input C from Step 5 of Figure 2 (the lift-reduction step) which is a full-precision basis with no special format into a valid input of `Lift-`$\widetilde{\mathrm{L}}^1$ which requires a truncated basis $B' \cdot E$. Just as in `Lift-`$\widetilde{\mathrm{L}}^1$ we use a stronger reduction parameter to compensate for needing a truncation.

Inputs: Valid LLL parameters $\Xi_1 > \Xi$; C Ξ-reduced with $\beta_k = \log \max \|C\|$; a lifting target ℓ_k;
Output: U unimodular, such that $\sigma_\ell CU$ is Ξ-reduced

1. $C'F := \mathrm{MSB}_{\ell_k + c_3 d}(C)$
2. Call `TrLiftLLL` on $(C'F, \Xi_1)$. Let $D^{-1}U_0 D$ be the output.
3. $B' := C'FD^{-1}U_0$; $E := D$
4. Call `Lift-`$\widetilde{\mathrm{L}}^1$ on (B', E, Ξ_1). Let U_{ℓ_k} be the output.
5. Return $U := D^{-1}U_0 D U_{\ell_k}$.

Fig. 5. From Figure 2 to Lift-$\widetilde{\mathrm{L}}^1$

This processing before `Lift-`$\widetilde{\mathrm{L}}^1$ is similar to what goes on inside of `Lift-`$\widetilde{\mathrm{L}}^1$. The accuracy follows from Lemma 3, Theorem 2, Theorem 5, and Lemma 7. While the complexity of this processing is necessarily less than the bit-complexity of `Lift-`$\widetilde{\mathrm{L}}^1$, $\mathcal{O}(d^{3+\varepsilon}(d+\ell_k+\tau_k)+d^\omega \mathcal{M}(\ell_k)\log \ell_k + \ell_k \log(\beta_k + \ell_k))$ from Theorem 6, which we can use as $\mathcal{C}_k$ from Lemma 6.

We now amortize the costs of all calls to Step 5 using Figure 5. More precisely, we bound $\sum_k \ell_k$ and $\sum_k \tau_k$ more tightly than using a generic bound for the ℓ_k's (resp. τ_k's). For the ℓ_k's, we have $\sum_k \ell_k \leq \log\det H \leq d\beta$. To handle the τ_k's, we adjust the standard LLL energy/potential analysis to allow for the small perturbations of $r_{i,i}$'s due to the various truncations.

Lemma 11. *Consider the execution of Steps 2–8 of* $\widetilde{\mathrm{L}}^1$ *(Figure 2). Let $H \in \mathbb{Z}^{d\times d}$ be the initial Hermite Normal Form. Let $\Xi_0 = (\delta_0, \eta_0, \theta_0)$ be the strongest set of LLL-parameters used within the execution. Let B be a basis occuring at any moment of Step 5 during the execution. Let R be the R-factor of B and n_{MSB} be the number of times* MSB *has been called so far. We define the energy of B as $\mathcal{E}(B, n_{\mathrm{MSB}}) := \frac{1}{\log 1/\delta_0}\left(\sum_i [(i-1)\cdot \log r_{i,i}] + d^2 n_{\mathrm{MSB}}\right)$ (using the natural logarithm). Then the number of LLL-switches performed so far satisfies $\tau \leq \mathcal{E}(B, n_{\mathrm{MSB}}) = \mathcal{O}(d \cdot \log\det H)$.*

Proof. The basis operations modifying the energy function are the LLL switches, the truncations (and returns from truncations), the adjunctions of a vector at Steps 3–4 of the algorithm from Figure 2 and the lifts. We show that any of these operations cannot decrease the energy function.

As Ξ_0 is the strongest set of LLL parameters ever considered during the execution of the algorithm, each LLL switch increases the weighted sum of the $r_{i,i}$'s (see [16, (1.23)]) and hence $\mathcal{E}$ by at least 1.

We now consider truncations. Each increase of n_{MSB} possibly decreases each $r_{i,i}$ (and again when we return from the truncation). We see from Lemma 1 and our choices of precisions p that for any two LLL parameters $\Xi' < \Xi$ there exists an $\varepsilon < 1$ such that each $r_{i,i}$ decreases by a factor no smaller than $(1+\varepsilon)$. Overall, the possible decrease of the weighted sum of the $r_{i,i}$'s is counterbalanced by the term "$d^2 n_{\text{MSB}}$" from the energy function, and hence $\mathcal{E}$ cannot decrease.

Now, the act of adjoining a new row in Figure 2 does not change the previous $r_{i,i}$'s but increases their weights. Since at the moment of an adjoining all $\log r_{i,i}$'s except possibly the first one are non-negative and since the weight of the first one is zero, Steps 3–4 cannot decrease $\mathcal{E}$.

Finally, each product by σ_ℓ (including those within the calls to `TrLiftLLL'`) cannot decrease any $r_{i,i}$, by Lemma 4.

To conclude, the energy never decreases and any switch increases it by at least 1. This implies that the number of switches is bounded by the growth $\mathcal{E}(B, n_{\text{MSB}}) - \mathcal{E}((h_{d,d}), 0)$. The initial value $\mathcal{E}((h_{d,d}), 0)$ of the energy is ≥ 0. Also, at the end of the execution, the term $\sum[(i-1)\log r_{i,i}]$ is $\mathcal{O}(\log\det H)$. As there are 5 calls to MSB in the algorithm from Figure 4 (including those contained in the calls to `TrLiftLLL'`), we can bound $d^2 n_{\text{MSB}}$ by $5d^2\sum_k(\ell_k/d) = 5\log\det H$. □

We obtain our main result by combining Theorems 5 and 6, and Lemma 11 to amortize the LLL-costs in Lemma 6 (we bound $\log\det H$ by $d\beta$).

Theorem 7. *Given as inputs Ξ and a matrix $B \in \mathbb{Z}^{d\times d}$ with $\max\|\mathbf{b}_j\| \leq 2^\beta$, the $\widetilde{\mathrm{L}}^1$ algorithm returns a Ξ-reduced basis of $L(B)$ within $\mathcal{O}(d^{5+\varepsilon}\beta + d^{\omega+1+\varepsilon}\beta^{1+\varepsilon})$ bit operations.*

Acknowledgements

Andrew Novocin and Damien Stehlé were partly funded by the LaRedA ANR project. Gilles Villard was partly funded by the Gecko ANR project and by a CNRS research collaboration grant to visit the MAGMA computational algebra group of the University of Sydney. Part of this work was done while Damien Stehlé was hosted by Macquarie University and the University of Sydney, whose hospitalities are gratefully acknowledged.

References

1. Karim Belabas. A relative van Hoeij algorithm over number fields. *Journal of Symbolic Computation*, 37(5):641–668, 2004.
2. X.-W. Chang, D. Stehlé, and G. Villard. Perturbation analysis of the QR Factor R in the context of LLL lattice basis reduction. To appear in Mathematics of Computation. HAL Report ensl-00529425, `http://prunel.ccsd.cnrs.fr/ensl-00529425/en`, École Normale Supérieure de Lyon, France, 2010.
3. D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10(4):233–260, 1997.
4. D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.
5. F. Eisenbrand. Short vectors of planar lattices via continued fractions. *Inf. Process. Lett.*, 79(3):121–126, 2001.
6. F. Eisenbrand. *50 Years of Integer Programming 1958-2008, From the Early Years to the State-of-the-Art*, chapter Integer Programming and Algorithmic Geometry of Numbers. Springer-Verlag, 2009.
7. F. Eisenbrand and G. Rote. Fast reduction of ternary quadratic forms. In *Proceedings of the 2001 Cryptography and Lattices Conference (CALC'01)*, volume 2146 of *Lecture Notes in Computer Science*, pages 32–44. Springer-Verlag, 2001.
8. N. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM Publications, 2002.

9. M. van Hoeij. Factoring polynomials and 0-1 vectors. In *Proceedings of the 2001 Cryptography and Lattices Conference (CALC'01)*, volume 2146 of *Lecture Notes in Computer Science*, pages 45–50. Springer-Verlag, 2001.
10. M. van Hoeij and A. Novocin. Gradual sub-lattice reduction and a new complexity for factoring polynomials. In *Proceedings of the 9th Latin American Theoretical Informatics Symposium LATIN 2010*, volume 6034 of *Lecture Notes in Computer Science*, pages 539–553. Springer-Verlag, 2010.
11. E. Kaltofen. On the complexity of finding short vectors in integer lattices. In *Proceedings of EUROCAL'83*, volume 162 of *Lecture Notes in Computer Science*, pages 236–244. Springer-Verlag, 1983.
12. R. Kannan, A. K. Lenstra, and L. Lovász. Polynomial factorization and nonrandomness of bits of algebraic and some transcendental numbers. In *Proceedings of STOC 1984*, pages 191–200. ACM Press, 1984.
13. D. Knuth. The analysis of algorithms. In *Actes du Congrès International des Mathématiciens (Nice, 1970)*, volume 3, pages 269–274. Gauthiers-Villars, 1971.
14. H. Koy and C. P. Schnorr. Segment LLL-reduction of lattice bases. In *Proceedings of the 2001 Cryptography and Lattices Conference (CALC'01)*, volume 2146 of *Lecture Notes in Computer Science*, pages 67–80. Springer-Verlag, 2001.
15. D. H. Lehmer. Euclid's algorithm for large numbers. *American Mathematical Monthly*, 45:227–233, 1938.
16. A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.
17. K. S. McCurley and J. L. Hafner. Asymptotically fast triangularization of matrices over rings. *SIAM Journal on Computing*, 20:1068–1083, 1991.
18. D. Micciancio and S. Goldwasser. *Complexity of lattice problems: a cryptographic perspective*. Kluwer Academic Press, 2002.
19. I. Morel, D. Stehlé, and G. Villard. From an LLL-reduced basis to another. In progress.
20. I. Morel, D. Stehlé, and G. Villard. H-LLL: using Householder inside LLL. In *Proceedings of the 2009 international symposium on Symbolic and algebraic computation (ISSAC'09)*, pages 271–278. ACM Press, 2009.
21. P. Q. Nguyen and D. Stehlé. An LLL algorithm with quadratic complexity. *SIAM Journal on Computing*, 39(3):874–903, 2009.
22. P. Q. Nguyen and B. Vallée (editors). *The LLL Algorithm: Survey and Applications*. Information Security and Cryptography. Springer-Verlag, 2009. Published after the LLL25 conference held in Caen in June 2007, in honour of the 25-th anniversary of the LLL algorithm.
23. A. Novocin. *Factoring Univariate Polynomials over the Rationals*. PhD thesis, Florida State University, 2008.
24. C. P. Schnorr. A more efficient algorithm for lattice basis reduction. *Journal of Algorithms*, 9(1):47–62, 1988.
25. C. P. Schnorr. Fast LLL-type lattice reduction. *Information and Computation*, 204:1–25, 2005.
26. A. Schönhage. Schnelle Berechnung von Kettenbruchentwicklungen. *Acta Informatica*, 1:139–144, 1971.
27. A. Schönhage. Factorization of univariate integer polynomials by Diophantine approximation and improved basis reduction algorithm. In *Proceedings of the 1984 International Colloquium on Automata, Languages and Programming (ICALP 1984)*, volume 172 of *Lecture Notes in Computer Science*, pages 436–447. Springer-Verlag, 1984.
28. A. Schönhage. Fast reduction and composition of binary quadratic forms. In *Proceedings of the 1991 International Symposium on Symbolic and Algebraic Computation (ISSAC'91)*, pages 128–133. ACM Press, 1991.
29. D. Stehlé. Floating-point LLL: theoretical and practical aspects. Chapter of [22].
30. A. Storjohann. Faster Algorithms for Integer Lattice Basis Reduction. Technical Report TR 249, ETH, Dpt. Comp. Sc., Zürich, Switzerland, 1996.
31. A. Storjohann and G. Labahn. Asymptotically fast computation of Hermite normal forms of integer matrices. In *Proceedings of the 1996 international symposium on Symbolic and algebraic computation (ISSAC'96)*, pages 259–266. ACM Press, 1996.
32. C. K. Yap. Fast unimodular reduction: planar integer lattices. In *Proceedings of the 1992 Symposium on the Foundations of Computer Science (FOCS 1992)*, pages 437–446. IEEE Computer Society Press, 1992.

Appendix 1 - Recipes used in the proof of Lemma 6

Let us first recall useful recipes for partially linearizing integer matrices and reducing the bit-cost of their products using asymptotically fast matrix multiplication algorithms. If one is interested in $\omega = 3$, then applying the naive matrix multiplication algorithm directly (without the linearization) already provides the given complexity upper bounds.

Recipe 1 *Let B and U be two $d \times d$ integer matrices such that $\sum_{i=1}^{d} \log \max_{1\le j\le d} |b_{i,j}|$ and $\sum_{j=1}^{d} \log \max_{1\le i\le d} |u_{i,j}|$ are both bounded by some S. We show how to compute the product $B \cdot U$ within $\mathcal{O}(d^{\omega}\mathcal{M}(S/d + \log d))$ bit operations.*

We reduce the product $B \cdot U$ to a product with balanced row and column bit-sizes by splitting into several rows the rows of B for which $\log \max_{1\le j\le d} |b_{i,j}| \ge \beta$, with $\beta := \lceil S/d \rceil$. We also split into several columns the columns of U for which $\log \max_{1\le i\le d} |u_{i,j}| \ge \beta$. More precisely, for $1 \le i \le d$, let $s_i = \lceil (\log \max_{1\le j\le d} |b_{i,j}|)/\beta \rceil$, and, for $1 \le j \le d$, let $t_j = \lceil (\log \max_{1\le i\le d} |u_{i,j}|)/\beta \rceil$. If $\mathbf{x}$ and $\mathbf{y}$ respectively denote row i of B and column j of U, then they are respectively replaced by

$$\begin{bmatrix} x_1^{(0)} & \cdots\cdots & x_d^{(0)} \\ \vdots & \cdots\cdots & \vdots \\ x_1^{(s_i-1)} & \cdots\cdots & x_d^{(s_i-1)} \end{bmatrix}$$

and

$$\begin{bmatrix} y_1^{(0)} & \cdots & y_1^{(t_j-1)} \\ \vdots & \cdots & \vdots \\ \vdots & \cdots & \vdots \\ y_d^{(0)} & \cdots & y_d^{(t_j-1)} \end{bmatrix},$$

where $x_k = \sum_{l=0}^{s_i-1} x_k^{(l)} 2^{l\beta}$, with $\log |x_k^{(l)}| \le \beta$, and $y_k = \sum_{l=0}^{t_j-1} y_k^{(l)} 2^{l\beta}$, with $\log |y_k^{(l)}| \le \beta$. The inner product $\mathbf{x} \cdot \mathbf{y}$ is then obtained by summing the entries of $D_1 P D_2$, where P is the product of the two matrices above (which are sub-matrices of the expansions of B and U), $D_1 := \mathrm{diag}_{l<s_i}(2^{l\beta})$, and $D_2 := \mathrm{diag}_{l<t_j}(2^{l\beta})$. Summing along antidiagonals and then summing the partial sums costs $\mathcal{O}(s_i t_j(\beta + \log d))$. The number of rows of the expansion of B is less than $\sum_i s_i \le d + \frac{d}{S}\sum_i \log \max_{1\le j\le d} |b_{i,j}| \le 2d$. Similarly, the number of columns of the expansion of U is less than $\sum_j t_j \le d + \frac{d}{S}\sum_j \log \max_{1\le i\le d} |u_{i,j}| \le 2d$. To complete the proof, note that all the entries of these expanded matrices have bit-lengths $\mathcal{O}(\beta)$. □

Recipe 2 *Let $k \le \log d$. Let U be a $d \times (d/2^k)$ integer matrix whose entries have bit-size $\le 2^k\gamma$, and B a $d \times d$ integer matrix such that $\sum_{j=1}^{d} \log \|\mathbf{b}_j\| \le d\gamma$, for some γ. Let $C = BU$ and assume that the entries of C have bit-size $\le 2^k\gamma$. We show how to compute C within $\mathcal{O}(d^{\omega+\varepsilon}\mathcal{M}(\gamma))$ bit operations, where ε is $o(1)$*

For $l \ge 0$ we see that B has at most $d/2^l$ columns $\mathbf{b}_j$ such that $\log \|\mathbf{b}_j\| \ge 2^l\gamma$. For $l > 0$, let J_l denote the set of the indices of the columns of B such that $2^l\gamma \le \log \|\mathbf{b}_j\| < 2^{l+1}\gamma$. Note that $J_l = \emptyset$ for $l > \log d$. We denote by J_0 the set of indices of the columns with $\log \|\mathbf{b}_j\| < 2\gamma$. For simplifying the cost bound discussion hereafter we assume that J_l has exactly $d/2^l$ elements (rather than $\le d/2^l$). Let also $B^{(l)}$ be the submatrix of B formed by the columns whose indices are in J_l. Accordingly, let $U^{(l)}$ be the submatrix of U formed by the rows whose indices are in J_l. Then we may compute $C = BU$ in $\log d$ products since (taking a symmetric modulo representation)

$$C = \sum_l B^{(l)} U^{(l)} \bmod 2^{2^{k+1}\gamma}. \tag{1}$$

For $k \le l$, the matrix $B^{(l)}$ has dimension $d \times (d/2^l)$, its entries may be taken modulo $2^{2^{k+1}\gamma}$ using $\mathcal{O}((d^2/2^l)\mathcal{M}(2^l\gamma))$ hence $\mathcal{O}(d^{2+\varepsilon}\mathcal{M}(\gamma))$ bit operations. The resulting matrix is seen as the concatenation of 2^l square row blocks of dimension $d/2^l$. The matrix $U^{(l)}$ has $d/2^l$ rows and $d/2^k \ge d/2^l$ columns. We may decompose $U^{(l)}$ into 2^{l-k} square column blocks with $d/2^l$ columns.

The product $B^{(l)}U^{(l)}$ in (1) can be done by blocks within $\mathcal{O}(2^l \times 2^{l-k} \times (d/2^l)^\omega \times \mathcal{M}(2^k\gamma))$ hence $\mathcal{O}(d^{\omega+\varepsilon}\mathcal{M}(\gamma))$ bit operations.

For $k > l$ we proceed as for Recipe 1 with $\beta := 2^l\gamma$ for expanding $U^{(l)}$ into a matrix with $(d/2^k)\cdot(2^{k-l})$ columns. Hence $B^{(l)}$ is $d \times d/2^l$, and the expansion of $U^{(l)}$ is square of dimension $d/2^l$. Both have entries of bit size $\mathcal{O}(2^l\gamma)$. By decomposing $B^{(l)}$ into 2^l square row blocks with $d/2^l$ rows, we can compute the product $B^{(l)}U^{(l)}$ in time $\mathcal{O}(2^l(d/2^l)^\omega\mathcal{M}(2^l\gamma+\log d))$ and hence $\mathcal{O}(d^{\omega+\varepsilon}\mathcal{M}(\gamma))$ bit operations. Overall, the cost for computing C using (1) is $\mathcal{O}(d^{\omega+\varepsilon}\mathcal{M}(\gamma))$. □

Recipe 3 *Let B, U and $C = BU$ be $d \times d$ integer matrices. Assume that there exists $s_1, \ldots, s_d$ such that $\log\|\mathbf{c}_j\|$, and $\log\|\mathbf{u}_j\|$ are $\leq s_j$, and $\sum_j \log\|\mathbf{b}_j\|$, and $\sum_j s_j$ are $\leq S$, for some S. We show how to compute the product C within $\mathcal{O}(d^{\omega+\varepsilon}\mathcal{M}(S/d))$ bit operations, where ε is $o(1)$.*

We apply to C the column decomposition seen in Recipe 2 for B. For $0 < k \leq \log d$, we let I_k denote the set of the indices of the columns of C such that $2^k S/d \leq \log\|\mathbf{c}_j\| < 2^{k+1}S/d$. We denote by I_0 the set of indices of the columns with $\log\|\mathbf{c}_j\| < 2S/d$. Let also $U^{(k)}$ be the submatrix of U formed by the columns whose indices are in I_k. As prior, the cardinality of I_k is at most $d/2^k$.

To compute C, it suffices to compute the $B \cdot U^{(k)}$'s, for $0 \leq k \leq \log d$. This can be done within $\mathcal{O}(d^{\omega+\varepsilon}\mathcal{M}(S/d))$ bit operations by using Recipe 2. Bounding the number of k's by $\mathcal{O}(\log d)$ allows us to complete the proof. □

Appendix 2 - Proof of Lemma 7

Lemma 12. *Let Ξ_1, Ξ_2, Ξ_3 be valid reduction parameters with $\Xi_3 > \Xi_2$. There exists a constant c_2 such that the following holds for any $\ell \geq 0$. Let $B \in \mathbb{R}^{d\times d}$ be Ξ_1-reduced, U such that $\sigma_\ell BU$ is Ξ_3-reduced and ΔB with $\max \frac{\|\Delta\mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-\ell-c_2\cdot d}$. Then $\sigma_\ell(B+\Delta B)U$ is Ξ_2-reduced.*

Proof. By Lemma 5, there exists a constant c such that for all i, j we have $|u_{j,i}| \leq 2^{c\cdot d}\frac{r'_{i,i}}{r_{j,j}}$, where R (resp. R') is the R-factor of B (resp. $C = \sigma_\ell BU$). Let $C + \Delta C = \sigma_\ell(B + \Delta B)U$. The norm of $\Delta\mathbf{c}_i = \sum_j u_{j,i}\sigma_\ell\Delta\mathbf{b}_j$ is $\leq \sum_j 2^{-p+\ell+c\cdot d}\frac{r'_{i,i}}{r_{j,j}}\|\mathbf{b}_j\| \leq d\alpha_1^d 2^{-p+\ell+c\cdot d}r'_{i,i}$, by Theorem 1 and with p such that $\max\frac{\|\Delta\mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-p}$. Furthermore, we have $\|\mathbf{c}_i\| \geq e'_{i,i}$. This gives $\max\frac{\|\Delta\mathbf{c}_i\|}{\|\mathbf{c}_i\|} \leq d\alpha_1^d 2^{-p+\ell+c\cdot d}$. By Lemma 3 (applied to C and $C + \Delta C$), there exists c' such that if $p \geq \ell + c'\cdot d$, then $C + \Delta C$ is Ξ_2-reduced. □

By combining Lemmata 12 and 3, we have that a reducing U can be found by working on a truncation of B.

Lemma 7. *Let Ξ_1, Ξ_2, Ξ_3 be valid reduction parameters with $\Xi_3 > \Xi_2$. There exists a constant c_3 such that the following holds for any $\ell \geq 0$. Let $B \in \mathbb{R}^{d\times d}$ be Ξ_1-reduced and ΔB be such that $\max\frac{\|\Delta\mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-\ell-c_3\cdot d}$. If $\sigma_\ell(B+\Delta B)U$ is Ξ_3-reduced for some U, then $\sigma_\ell BU$ is Ξ_2-reduced.*

Proof. Let $\Xi_0 < \Xi_1$ be a valid set of reduction parameters. By Lemma 3, there exists a constant c such that if $\max\frac{\|\Delta\mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq 2^{-c\cdot d}$, then $B + \Delta B$ is non-singular and Ξ_0-reduced. We conclude by using Lemma 12. □

Appendix 3 - Proof of Theorem 2 and description of Algorithm `TrLiftLLL`

Theorem 2. *For any valid parameters $\Xi_1 < \Xi_2$ and constant c_4, there exists a constant c'_4 and an algorithm `TrLiftLLL` with the following specifications. It takes as inputs $\ell \geq 0$, $B \in$*

$\mathbb{Z}^{d\times d}$ and $E = \mathrm{diag}(2^{e_i})$ with $\max\|\mathbf{b}_i\| \leq 2^{c_4(\ell+d)}$, $e_i \in \mathbb{Z}$ and BE is Ξ_1-reduced; It runs in time $O(d^{2+\varepsilon}(d+\ell)(d+\ell+\tau) + d^2\log\max(1+|e_i|))$, where $\tau = O(d^2(\ell+d))$ is the number of switches performed during the single call it makes to H-LLL; And it returns two matrices U and D such that:

1. *$D = \mathrm{diag}(2^{d_i})$ with $d_i \in \mathbb{Z}$ satisfying $\max|e_i - d_i| \leq c_4'(\ell+d)$,*
2. *U is unimodular and $\max|u_{i,j}| \leq 2^{\ell + c_4'\cdot d}$,*
3. *$D^{-1}UD$ is unimodular and $\sigma_\ell(BE)(D^{-1}UD)$ is Ξ_2-reduced.*

The possible unbalancedness of the columns of BE (due to E), prevents us from applying H-LLL directly on $C = \sigma_\ell BE$. Indeed, even if we were dividing the full matrix by a large common power of 2, the resulting basis may have a bit-size that is arbitrarily large compared to d and ℓ. Our goal is to call H-LLL on a integral matrix whose entries have bit-sizes $\mathcal{O}(d+\ell)$. To circumvent the possible unbalanced-ness of the columns of C, we find blocks of consecutive vectors whose $r_{i,i}^{(C)}$'s have similar magnitudes, where $R^{(C)}$ is the R-factor of C, and we apply a column-scaling to re-balance C before calling H-LLL.

Finding blocks. The definition of block is motivated by Property (P) above. To determine meaningful blocks, the first step is to find good approximations to the $r_{i,i}^{(C)}$'s and $r_{i,i}^{(BE)}$'s (where $R^{(BE)}$ is the R-factor of BE). Computing the R-factor of a non-singular matrix is most often done by applying Householder's algorithm (see [8, Ch. 19]). The following lemma is a rigorous and explicit variant of standard backward stability results.

Lemma 13 ([2, Se. 6]). *Let $p \geq 0$ and $B \in \mathbb{R}^{d\times d}$ be non-singular with R-factor R. Let $\widehat{R}$ be the R-factor computed by Householder's algorithm with floating-point precision p. If $c_5 2^{-p} < 1$ with $c_5 = 80d^2$, then there exists an orthogonal $\widehat{Q}$ such that $\widehat{Q}\widehat{R} = B + \Delta B$ with $\max\frac{\|\Delta\mathbf{b}_i\|}{\|\mathbf{b}_i\|} \leq c_5 2^{-p}$.*

By Lemma 2, we have that $\mathrm{cond}(R^{(BE)}) \leq \frac{\rho+1}{\rho-1}\rho^d$. Since $R^{(BE)} = R^{(B)}\cdot E$, with $R^{(B)}$ the R-factor of B, we have $\mathrm{cond}(R^{(B)}) \leq \frac{\rho+1}{\rho-1}\rho^d$ (because $\mathrm{cond}(\cdot)$ is invariant under column scaling). Now, by Lemmata 1 and 13, for any c there exists c' such that Householder's algorithm with precision $p = c'd$ allows us to find $\widehat{R}^{(B)}$ with $\max\frac{\|\widehat{\mathbf{r}}_i^{(B)} - \mathbf{r}_i^{(B)}\|}{\|\mathbf{r}_i^{(B)}\|} \leq 2^{-cd}$. By defining $\widehat{R}^{(BE)}$ by $\widehat{R}^{(B)}\cdot E$, we have $\max\frac{\|\widehat{\mathbf{r}}_i^{(BE)} - \mathbf{r}_i^{(BE)}\|}{\|\mathbf{r}_i^{(BE)}\|} \leq 2^{-cd}$. The latter can be made $\leq \frac{1}{100}$.

We now show that we can also compute approximations to the $r_{i,i}^{(C)}$'s. Let $B = Q^{(B)}R^{(B)}$ and $\sigma_\ell B = Q^{(\sigma_\ell B)}R^{(\sigma_\ell B)}$ be the QR factorizations of B and $\sigma_\ell B$ respectively. We have:

$$\begin{aligned}
\mathrm{cond}(R^{(\sigma_\ell B)}) &= \left\| |R^{(\sigma_\ell B)}||(R^{(\sigma_\ell B)})^{-1}| \right\| \\
&= \left\| |(Q^{(\sigma_\ell B)})^t \sigma_\ell Q^{(B)} R^{(B)}||(R^{(B)})^{-1}(Q^{(B)})^t \sigma_\ell^{-1} Q^{(\sigma_\ell B)}| \right\| \\
&\leq \left\| |(Q^{(\sigma_\ell B)})^t|\sigma_\ell|Q^{(B)}||R^{(B)}||(R^{(B)})^{-1}||(Q^{(B)})^t|\sigma_\ell^{-1}|Q^{(\sigma_\ell B)}| \right\| \\
&\leq d^2 2^\ell \mathrm{cond}(R^{(B)}) = d^2 2^\ell \mathrm{cond}(R^{(B)}E).
\end{aligned}$$

Since $R^{(B)}E$ is the R-factor of BE which is reduced, Lemma 2 gives that $\mathrm{cond}(R^{(\sigma_\ell B)}) \leq d^2\frac{\rho+1}{\rho-1}\rho^d 2^\ell$. Now, Lemmata 1 and 13 imply that for any c there exists c' such that Householder's algorithm with precision $p = 2\ell + c'd$ allows us to find $\widehat{R}^{(\sigma_\ell B)}$ with $\max\frac{\|\widehat{\mathbf{r}}_i^{(\sigma_\ell B)} - \mathbf{r}_i^{(\sigma_\ell B)}\|}{\|\mathbf{r}_i^{(\sigma_\ell B)}\|} \leq$

$2^{-\ell-cd}$. Since $\|\mathbf{r}_i^{(\sigma_\ell B)}\| \leq 2^\ell \|\mathbf{r}_i^{(B)}\| \leq 2^\ell \alpha^d r_{i,i}^{(B)} \leq 2^\ell \alpha^d r_{i,i}^{(\sigma_\ell B)}$ (using Theorem 1 and Lemma 4), we obtain that Householder's algorithm with precision $2\ell + \mathcal{O}(d)$ provides some $\widehat{r}_{i,i}^{(\sigma_\ell B)}$'s such that $\max \frac{|\widehat{r}_{i,i}^{(\sigma_\ell B)} - r_{i,i}^{(\sigma_\ell B)}|}{r_{i,i}^{(\sigma_\ell B)}} \leq \frac{1}{100}$. Since $R^{(C)} = R^{(\sigma_\ell B)} E$, we have $\max \frac{|\widehat{r}_{i,i}^{(C)} - r_{i,i}^{(C)}|}{r_{i,i}^{(C)}} \leq \frac{1}{100}$, with $\widehat{R}^{(C)} = \widehat{R}^{(\sigma_\ell B)} E$. Furthermore, as the run-time of Householder's algorithm in precision p is $\mathcal{O}(d^3 p^{1+\varepsilon})$, the computation of these $\widehat{r}_{i,i}^{(C)}$'s costs $\mathcal{O}(d^3(\ell+d)^{1+\varepsilon})$.

We define the blocks of vectors of C as follows: The first block starts with $\mathbf{c}_{i_1} = \mathbf{c}_1$ and stops with $\mathbf{c}_{i_2-1}$ where i_2 is the smallest i such that $\min_{j \geq i} \widehat{r}_{j,j}^{(C)} > \nu \cdot \max_{j<i} \widehat{r}_{j,j}^{(C)}$ (if $i_2 = d+1$, then the process ends); The kth block starts with $\mathbf{c}_{i_k}$ and stops with $\mathbf{c}_{i_{k+1}-1}$ where i_{k+1} is the smallest index $i > i_k$ such that $\min_{j \geq i} \widehat{r}_{j,j}^{(C)} > \nu \cdot \max_{j<i} \widehat{r}_{j,j}^{(C)}$. The purpose of the constant $\nu \geq 4$, to be set later, is to handle the inaccuracy of $\widehat{R}^{(C)}$ and to ensure that the matrix $CD^{-1}UD$ eventually obtained by `TrLiftLLL` will be size-reduced.

Let $I_k = [i_k, i_{k+1})$. Since $\nu \geq 4$, Property (P) implies that if we were to call H-LLL on C, the unimodular U that we would obtain would satisfy $u_{i,j} = 0$ if $i \in I_{k_1}$ and $j \in I_{k_2}$ with $k_1 < k_2$, i.e., U would be (I_k)-block upper triangular. Any diagonal block-submatrix of U would be unimodular. Computing the I_k's from the $\widehat{r}_{j,j}$'s may be done in time $\mathcal{O}(d^2(d + \ell + \log \max(1 + |e_i|)))$.

By construction of the blocks, the amplitude of $r_{i,i}^{(C)}$'s within a block is bounded.

Lemma 14. *We use the same notations as above. We let $(\ell_i = r_{i,i}^{(C)}/r_{i,i}^{(BE)}$. There exists a constant c_6 (depending on Ξ_1 and ν only) such that for any k, we have $\frac{\max_{i \in I_k} r_{i,i}^{(C)}}{\min_{i \in I_k} r_{i,i}^{(C)}} \leq 2^{c_6 |I_k|} \cdot \max_{i \in I_k} \ell_i$.*

Proof. Let $i, j \in I_k$. We are to compute an upper bound for $\frac{r_{j,j}^{(C)}}{r_{i,i}^{(C)}}$. If $j \leq i$, the reducedness of BE implies that $\frac{r_{j,j}^{(C)}}{\ell_j} \leq \alpha^{i-j} \frac{r_{i,i}^{(C)}}{\ell_i}$, for α as in Theorem 1. The fact that $\ell_i \geq 1$ (see Lemma 4) provides the result. Assume now that $j > i$. If $r_{i,i}^{(C)} = \max_{t \geq i} r_{t,t}^{(C)}$, then the bound holds. Otherwise, by definition of the blocks, there exists $i' > i$ in I_k such that $r_{i',i'}^{(C)} \leq 2\nu \cdot r_{i,i}^{(C)}$ (the factor 2 takes the inaccuracy of $\widehat{R}$ into account). By induction, it can be shown that $r_{i'',i''}^{(C)} \leq (2\nu)^{|I_k|} r_{i,i}^{(C)}$, with $i'' = i_{k+1} - 1$. We conclude that $\frac{r_{j,j}^{(C)}}{r_{i,i}^{(C)}} \leq (2\nu)^{|I_k|} \frac{r_{j,j}^{(C)}}{r_{i'',i''}^{(C)}} \leq (2\nu\alpha)^{|I_k|} \ell_j$, by using the first part of the proof (since $j \leq i''$). □

Re-balancing the columns of C. The blocks allow us to define the diagonal matrix D of Theorem 2. We define the gap between two blocks I_k and I_{k+1} to be $g_k = \frac{\min_{j \in I_{k+1}} \widehat{r}_{j,j}^{(BE)}}{\max_{j \in I_k} \widehat{r}_{j,j}^{(C)}}$. We define $D = \mathrm{diag}(2^{d_i})$ such that the block structure is preserved, but the gaps get shrunk: For $i \in I_k$, we set $d_i = e_1 + \sum_{k'<k} \lceil \log_2 g_{k'} / \sqrt{\nu} \rceil$.

We prove several facts about this scaling.

(*i*) The matrix $B' = BED^{-1}$ is Ξ_1-reduced, because $r_{j,j}^{(C)} \geq r_{j,j}^{(BE)}$ for all j.

(*ii*) The matrix $C' = CD^{-1}$ with R-factor $R^{(C')} = R^{(C)} D^{-1}$ admits the same block-structure a C: For any k, we have $\min_{j \in I_{k+1}} r_{j,j}^{(C')} \geq \nu' \cdot \max_{j \in I_k} r_{j,j}^{(C')}$, with $\nu' = \sqrt{\nu}/2 \geq 1$.

(*iii*) The d_i's satisfy Property 1 of Theorem 2: Thanks to the reducedness of BE, the size condition on B, and Lemma 4, each e_i is within $\mathcal{O}(\ell + d)$ of $\log r_{i,i}^{(C)}$. Thanks to Lemmata 14 and 4 (in particular the fact that the product of all ℓ_j's is 2^ℓ), the same holds for the d_i's.

LLL-reducing. We now call H-LLL on input matrix C', with LLL-parameters $\Xi > \Xi_2$, and let $C^{(2)}$ be the output matrix. Thanks to (iii), the matrix C' belongs to $2^{-c(\ell+d)}\mathbb{Z}^{d\times d}$ for some constant c, and each $c'_{i,j}$ may be stored on $\mathcal{O}(\ell+d)$ bits. I.e., the matrix C' is balanced. As a consequence, the call to H-LLL costs $\mathcal{O}(d^{2+\varepsilon}(d+\ell+\tau)(d+\ell))$ bit operations (see [20, Th. 4.4]), where τ be the number of switches performed.

Let U be the corresponding unimodular transform (which can be recovered from C' and $C^{(2)}$ by a matrix inversion, costing $\mathcal{O}(d^3(d+\ell)^{1+\varepsilon})$). Lemma 5 and the fact that B' is Ξ_1-reduced (by (i)) ensure that Property 2 of Theorem 2 is satisfied. Also, since C' follows the block-structure defined by the I_k's (by (ii)), Property (P) may be used to assert that U is $(I_k)_k$-block upper triangular and that its diagonal blocks are unimodular. The coefficients of D are non-decreasing, and they are constant within any I_k. This ensures that $D^{-1}UD$ is integral and that its diagonal blocks are exactly those of U, and thus that $D^{-1}UD$ is unimodular.

Let $C^{(3)} = \sigma_\ell BED^{-1}UD = C^{(2)}D$. It remains to show that $C^{(3)}$ is Ξ_2-reduced. Let $R^{(2)}$ (resp. $R^{(3)}$) be the R-factor of $C^{(2)}$ (resp. $C^{(3)}$). Let $\Xi = (\delta, \eta, \theta)$ and $\Xi_2 = (\delta_2, \eta_2, \theta_2)$. If i and j belong to the same I_k, then $|r^{(3)}_{i,j}| \leq \eta r^{(3)}_{i,i} + \theta r^{(3)}_{j,j}$, because this holds for $R^{(2)}$ and $\frac{r^{(3)}_{i,j}}{r^{(2)}_{i,j}} = \frac{r^{(3)}_{i,i}}{r^{(2)}_{i,i}} = \frac{r^{(3)}_{j,j}}{r^{(2)}_{j,j}} = 2^{d_{i_k}}$. Since $\eta < \eta_2$ and $\theta < \theta_2$, the size-reduction condition for (i,j) is satisfied. Similarly, the Lovász conditions are satisfied inside the I_k's. They are also satisfied for any $i = i_k - 1$, since $\mathbf{c}^{(2)}_{i_k}$ is multiplied by $2^{d_{i_k}} \geq 2^{d_{i_k-1}}$. It remains to check the size-reduction conditions for (i,j) with $i \in I_k$, $j \in I_{k'}$ and $k' > k$. By reducedness of $C^{(2)}$, we have $|r^{(2)}_{i,j}| \leq \eta r^{(2)}_{i,i} + \theta r^{(2)}_{j,j}$. Since it was the case for R', by Property (P), we have that $r^{(2)}_{i,i} \leq \frac{1}{\nu'} r^{(2)}_{j,j}$ (with $\nu' = \sqrt{\nu}/2$), and thus $|r^{(2)}_{i,j}| \leq (\theta + \frac{1}{\nu'}) r^{(2)}_{j,j}$. This gives $|r^{(3)}_{i,j}| \leq (\theta + \frac{1}{\nu'}) r^{(3)}_{j,j}$. In order to ensure size-reducedness, it thus suffices to choose ν such that $\theta + \frac{1}{\nu'} \leq \theta_2$. □

Improved Analysis of Kannan's Shortest Lattice Vector Algorithm
(Extended Abstract)

Guillaume Hanrot[1,⋆] and Damien Stehlé[2]

[1] LORIA/INRIA Lorraine, Technopôle de Nancy-Brabois,
615 rue du jardin botanique, F-54602 Villers-lès-Nancy Cedex, France
hanrot@loria.fr
http://www.loria.fr/~hanrot
[2] CNRS and ÉNS Lyon/ LIP, 46 allée d'Italie, 69364 Lyon Cedex 07, France
damien.stehle@ens-lyon.fr
http://perso.ens-lyon.fr/damien.stehle

Abstract. The security of lattice-based cryptosystems such as NTRU, GGH and Ajtai-Dwork essentially relies upon the intractability of computing a shortest non-zero lattice vector and a closest lattice vector to a given target vector in high dimensions. The best algorithms for these tasks are due to Kannan, and, though remarkably simple, their complexity estimates have not been improved since over twenty years. Kannan's algorithm for solving the shortest vector problem (SVP) is in particular crucial in Schnorr's celebrated block reduction algorithm, on which rely the best known generic attacks against the lattice-based encryption schemes mentioned above. In this paper we improve the complexity upper-bounds of Kannan's algorithms. The analysis provides new insight on the practical cost of solving SVP, and helps progressing towards providing meaningful key-sizes.

1 Introduction

A lattice L is a discrete subgroup of some $\mathbb{R}^n$. Such an object can always be represented as the set of integer linear combinations of at most n vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d$. These vectors can be chosen linearly independent, and in that case, we say that they are a basis of the lattice L. The most famous algorithmic problem associated with lattices is the so-called shortest vector problem (SVP). Its computational variant is to find a non-zero lattice vector of smallest Euclidean length — this length being the minimum $\lambda(L)$ of the lattice — given a basis of the lattice. Its decisional variant is known to be NP-hard under randomised reductions [2], even if one only asks for a vector whose length is no more than $2^{(\log d)^{1-\epsilon}}$ times the length of a shortest vector [12] (for any $\epsilon > 0$).

SVP is of prime importance in cryptography since a now quite large family of public-key cryptosystems relies more or less on it. The Ajtai-Dwork cryptosystem [4] relies on d^c-SVP for some $c > 0$, where $f(d)$-SVP is the problem of finding

⋆ Work partially supported by CNRS GDR 2251 "Réseau de théorie des nombres".

A. Menezes (Ed.): CRYPTO 2007, LNCS 4622, pp. 170–186, 2007.

the shortest non-zero vector in the lattice L, under the promise that any vector of length less than $f(d) \cdot \lambda(L)$ is parallel to it. The GGH cryptosystem [11] relies on special instances of the Closest Vector Problem (CVP), a non-homogeneous version of SVP. Both the Ajtai-Dwork and GGH cryptosystems have been shown impractical for real-life parameters [25,23] (the initial GGH containing a major theoretical flaw as well). Finally, one strongly suspects that in NTRU [15] the private key can be read on the coordinates of a shortest vector of the Coppersmith-Shamir lattice [8]. The best known generic attacks against these encryption schemes are based on solving SVP. It is therefore highly important to know precisely what complexity is achievable, both in theory and practice, in particular to select meaningful key-sizes. Most often, for cryptanalysing lattice-based cryptosystems, one considers Schnorr's block-based algorithms [28, 30], such as BKZ. These algorithms internally solve instances of SVP in much lower dimensions (related to the size of the block). They help solving relaxed variants of SVP in high dimensions. Increasing the dimensions up to which one can solve SVP helps decreasing the relaxation factors that are achievable in higher dimensions. Solving the instances of SVP is the computationally expensive part of the block-based reduction algorithms.

Two main algorithms are known for solving SVP. The first one is based on the deterministic exhaustive enumeration of lattice points within a small convex body. It is known as Fincke-Pohst's enumeration algorithm [9] in the algorithmic number theory community. Cryptographers know it as Kannan's algorithm [16]. There are two main differences between both: firstly, in Kannan's algorithm, a long pre-computation on the basis is performed before starting the enumeration process; secondly, Kannan enumerates integer points in a hyper-parallelepiped whereas Fincke and Pohst consider an hyper-ellipsoid which is strictly contained in Kannan's hyper-parallelepiped – though Kannan may have chosen the hyper-parallelepiped in order to simplify the complexity analysis. Kannan obtained a $d^{d+o(d)}$ complexity bound (in the complexity bounds mentioned in the introduction, there is an implicit factor that is polynomial in the bit-size of the input). In 1985, Helfrich [13] refined Kannan's analysis, and obtained a $d^{d/2+o(d)}$ complexity bound. On the other hand, Ajtai, Kumar and Sivakumar [5] designed a probabilistic algorithm of complexity $2^{O(d)}$. The best exponent constant is likely to be small, as suggested by some recent progress [26]. A major drawback of this algorithm is that it requires an exponential space, whereas Kannan's requires a polynomial space.

Our main result is to lower Helfrich's complexity bound on Kannan's algorithm, from $d^{\frac{d}{2}+o(d)} \approx d^{0.5 \cdot d}$ to $d^{\frac{d}{2e}+o(d)} \approx d^{0.184 \cdot d+o(d)}$. This may explain why Kannan's algorithm is tractable even in moderate dimensions. Our analysis can also be adapted to Kannan's algorithm for CVP: it decreases Helfrich's complexity bound from $d^{d+o(d)}$ to $d^{d/2+o(d)}$. The complexity improvement for SVP provides better worst-case efficiency/quality trade-offs for Schnorr's block-based algorithms [28, 30, 10].

It must be noted that if one follows our analysis step by step, the derived $o(d)$ may be large when evaluated for some practical d. The hidden constants can be improved (for some of them it may be easy, for others it is probably much harder).

No attempt was made to improve them and we believe that it would have complicated the proof with irrelevant details. In fact, most of our analysis consists in estimating the number of lattice points within convex bodies and showing that the approximations by the volumes are almost valid. By replacing this discretisation by heuristic volume estimates, one obtains very small hidden constants.

Our complexity improvement is based on a fairly simple idea. It is equivalent to generate all lattice points within a ball and to generate all integer points within an ellipsoid (consider the ellipsoid defined by the quadratic form naturally associated with the given lattice basis). Fincke and Pohst noticed that it was more efficient to work with the ellipsoid than to consider a parallelepiped containing it: indeed, when the dimension increases, the ratio between the two volumes tends to 0 very quickly. In his analysis, instead of considering the ellipsoid, Kannan bounds the volume of the parallelepiped. Using rather involved technicalities, we bound the number of points within related ellipsoids. Some parts of our proof could be of independent interest. For example, we show that for any Hermite-Korkine-Zolotarev-reduced (HKZ-reduced for short) lattice basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$, and any subset I of $\{1, \ldots, d\}$, we have:

$$\frac{\|\boldsymbol{b}_1\|^{|I|}}{\prod_{i\in I}\|\boldsymbol{b}_i^*\|} \leq \sqrt{d}^{|I|\left(1+\log\frac{d}{|I|}\right)},$$

where $(\boldsymbol{b}_i^*)_{i\leq d}$ is the Gram-Schmidt orthogonalisation of the $\boldsymbol{b}_i$'s. This generalises the results of [28] on the quality of HKZ-reduced bases.

Practical Implications. We do not change Kannan's algorithm, but only improve its complexity upper-bound. As a consequence, the running-time of Kannan's algorithm remains the same. Nevertheless, our work may still have some important practical impact. First of all, it revives the interest on Kannan's algorithm. Surprisingly, although it has the best complexity upper-bound, it is not the one implemented in the usual number theory libraries (e.g., NTL [32] and Magma [18] implement Schnorr-Euchner's variant [30]): we show that by using Kannan's principle (i.e., pre-processing the basis before starting the enumeration), one can solve SVP in larger dimensions. This might point a problem in NTRU's security estimates, since they are derived from experimentations with NTL. Secondly, our analysis helps providing a heuristic measure of the (practical) cost of solving SVP for a particular instance, which is both efficiently computable and reliable: given a lattice basis, it provides very quickly a heuristic upper bound on the cost of finding a shortest vector.

Road-Map of the Paper. In Section 2, we recall some basic definitions and properties on lattice reduction. Section 3 is devoted to the description of Kannan's algorithm and Section 4 to its complexity analysis. In Section 5, we give without much detail our sibling result on CVP, as well as direct consequences of our result for block-based algorithms. In Section 6, we discuss the practical implications of our work.

Notation. All logarithms are natural logarithms, i.e., $\log(e) = 1$. Let $\|\cdot\|$ and $\langle\cdot,\cdot\rangle$ be the Euclidean norm and inner product of $\mathbb{R}^n$. Bold variables are vectors. We

use the bit complexity model. The notation $\mathcal{P}(n_1, \ldots, n_i)$ means $(n_1 \cdot \ldots \cdot n_i)^c$ for some constant $c > 0$. If x is real, we denote by $\lfloor x \rceil$ a closest integer to it (with any convention for making it unique) and we define the centred fractional part $\{x\}$ as $x - \lfloor x \rceil$. Finally, for any integers a and b, we define $[\![a, b]\!]$ as $[a, b] \cap \mathbb{Z}$.

2 Background on Lattice Reduction

We assume that the reader is familiar with the geometry of numbers and its algorithmic aspects. Introductions may be found in [21] and [27].

Lattice Invariants. Let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d$ be linearly independent vectors. Their *Gram-Schmidt orthogonalisation* (GSO) $\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_d^*$ is the orthogonal family defined recursively as follows: the vector $\boldsymbol{b}_i^*$ is the component of $\boldsymbol{b}_i$ which is orthogonal to the span of the vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-1}$. We have $\boldsymbol{b}_i^* = \boldsymbol{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \boldsymbol{b}_j^*$ where $\mu_{i,j} = \frac{\langle \boldsymbol{b}_i, \boldsymbol{b}_j^* \rangle}{\|\boldsymbol{b}_j^*\|^2}$. For $i \leq d$ we let $\mu_{i,i} = 1$. Notice that the GSO family depends on the order of the vectors. If the $\boldsymbol{b}_i$'s are integer vectors, the $\boldsymbol{b}_i^*$'s and the $\mu_{i,j}$'s are rational. The *volume* of a lattice L is defined as $\det(L) = \prod_{i=1}^{d} \|\boldsymbol{b}_i^*\|$, where the $\boldsymbol{b}_i$'s are any basis of L. It does not depend on the choice of the basis of L and can be interpreted as the geometric volume of the parallelepiped naturally spanned by the basis vectors. Another important lattice invariant is the minimum. The *minimum* $\lambda(L)$ is the length of a shortest non-zero lattice vector.

The most famous lattice problem is the *shortest vector problem* (SVP). Here is its computational variant: given a basis of a lattice L, find a lattice vector whose norm is exactly $\lambda(L)$. The *closest vector problem* (CVP) is a non-homogeneous variant of SVP. We give here its computational variant: given a basis of a lattice L and a target vector in the real span of L, find a vector of L which is closest to the target vector.

The volume and the minimum of a lattice cannot behave independently. Hermite [14] was the first to bound the ratio $\frac{\lambda(L)}{(\det L)^{1/d}}$ as a function of the dimension only. His bound was later on greatly improved by Minkowski in his *Geometrie der Zahlen* [22]. *Hermite's constant* γ_d is defined as the supremum over d-dimensional lattices L of $\frac{\lambda(L)^2}{(\det L)^{2/d}}$. We have $\gamma_d \leq \frac{d+4}{4}$ (see [19]), which we will refer to as *Minkowski's theorem*.

Lattice Reduction. In order to solve lattice problems, a classical strategy consists in considering a lattice basis and trying to improve its quality (e.g., the slow decrease of the $\|\boldsymbol{b}_i^*\|$'s). This is called *lattice reduction*. The most usual notions of reduction are probably L^3 and HKZ. HKZ-reduction is very strong, but expensive to compute. On the contrary, L^3-reduction is fairly cheap, but an L^3-reduced basis is of much lower quality.

A basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ is *size-reduced* if its GSO family satisfies $|\mu_{i,j}| \leq 1/2$ for all $1 \leq j < i \leq d$. A basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ is said to be *Hermite-Korkine-Zolotarev-reduced* if it is size-reduced, the vector $\boldsymbol{b}_1$ reaches the lattice minimum, and the projections of the $(\boldsymbol{b}_i)_{i \geq 2}$'s orthogonally to the vector $\boldsymbol{b}_1$ are themselves an

HKZ-reduced basis. Lemma 1 immediately follows from this definition and Minkowski's theorem. It is the sole property on HKZ-reduced bases that we will use.

Lemma 1. *If $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ is HKZ-reduced, then for any $i \leq d$, we have:*

$$\|\boldsymbol{b}_i^*\| \leq \sqrt{\frac{d-i+5}{4}} \cdot \left(\prod_{j \geq i} \|\boldsymbol{b}_j^*\| \right)^{\frac{1}{d-i+1}}.$$

A basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ is *L^3-reduced* [17] if it is size-reduced and if its GSO satisfies the $(d-1)$ Lovász conditions: $\frac{3}{4} \cdot \left\|\boldsymbol{b}_{\kappa-1}^*\right\|^2 \leq \left\|\boldsymbol{b}_\kappa^* + \mu_{\kappa,\kappa-1}\boldsymbol{b}_{\kappa-1}^*\right\|^2$. The L^3-reduction implies that the norms of the GSO vectors never drop too fast: intuitively, the vectors are not far from being orthogonal. Such bases have useful properties, like providing exponential approximations to SVP and CVP. In particular, their first vector is relatively short.

Theorem 1 ([17]). *Let $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ be an L^3-reduced basis of a lattice L. Then we have $\|\boldsymbol{b}_1\| \leq 2^{\frac{d-1}{4}} \cdot (\det L)^{1/d}$. Moreover, there exists an algorithm that takes as input any set of integer vectors and outputs in deterministic polynomial time an L^3-reduced basis of the lattice they span.*

In the following, we will also need the fact that if the set of vectors given as input to the L^3 algorithm starts with a shortest non-zero lattice vector, then this vector is not changed during the execution of the algorithm: the output basis starts with the same vector.

3 Kannan's SVP Algorithm

Kannan's SVP algorithm [16] relies on multiple calls to the so-called short lattice points enumeration procedure. The latter finds all vectors of a given lattice that are in the sphere centred in $\mathbf{0}$ and of some prescribed radius. Variants of the enumeration procedure are described in [1].

3.1 Short Lattice Points Enumeration

Let $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ be a basis of a lattice $L \subset \mathbb{Z}^n$ and let $A \in \mathbb{Z}$. Our goal is to find all lattice vectors $\sum_{i=1}^d x_i \boldsymbol{b}_i$ of squared Euclidean norm $\leq A$. The enumeration works as follows. Suppose that $\|\sum_i x_i \boldsymbol{b}_i\|^2 \leq A$ for some integers x_i's. Then, by considering the components of the vector $\sum_i x_i \boldsymbol{b}_i$ on each of the $\boldsymbol{b}_i^*$'s, we obtain d equations:

$$\begin{aligned} (x_d)^2 \cdot \|\boldsymbol{b}_d^*\|^2 &\leq A, \\ (x_{d-1} + \mu_{d,d-1} x_d)^2 \cdot \|\boldsymbol{b}_{d-1}^*\|^2 &\leq A - (x_d)^2 \cdot \|\boldsymbol{b}_d^*\|^2, \\ &\ldots \end{aligned}$$

$$\left(x_i + \sum_{j=i+1}^{d} \mu_{j,i} x_j\right)^2 \cdot \|\boldsymbol{b}_i^*\|^2 \leq A - \sum_{j=i+1}^{d} l_j,$$
$$\ldots$$

where $l_i = (x_i + \sum_{j>i} x_j \mu_{j,i})^2 \cdot \|\boldsymbol{b}_i^*\|^2$. The algorithm of Figure 1 mimics the equations above. It can be shown that the bit-cost of this algorithm is bounded by the number of loop iterations times a polynomial in the bit-size of the input. We will prove that if the input basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ is sufficiently reduced and if $A = \|\boldsymbol{b}_1\|^2$, there are $\leq d^{\frac{d}{2e}+o(d)}$ loop iterations.

Input: An integer lattice basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$, a bound $A \in \mathbb{Z}$.
Output: All vectors in $L(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ that are of squared norm $\leq A$.
1. Compute the rational $\mu_{i,j}$'s and $\|\boldsymbol{b}_i^*\|^2$'s.
2. $\boldsymbol{x}$:=$\boldsymbol{0}$, $\boldsymbol{l}$:=$\boldsymbol{0}$, S:=$\emptyset$.
3. i:=1. While $i \leq d$, do
4. $\quad l_i$:=$(x_i + \sum_{j>i} x_j \mu_{j,i})^2 \|\boldsymbol{b}_i^*\|^2$.
5. $\quad$ If $i = 1$ and $\sum_{j=1}^{d} l_j \leq A$, then S:=$S \cup \{\sum_{j=1}^{d} x_j \boldsymbol{b}_j\}$, x_1:=$x_1 + 1$.
6. $\quad$ If $i \neq 1$ and $\sum_{j \geq i} l_j \leq A$, then
7. $\quad\quad i$:=$i - 1$, x_i:=$\left\lceil -\sum_{j>i}(x_j \mu_{j,i}) - \sqrt{\frac{A - \sum_{j>i} l_j}{\|\boldsymbol{b}_i^*\|^2}} \right\rceil$.
8. $\quad$ If $\sum_{j \geq i} l_j > A$, then i:=$i + 1$, x_i:=$x_i + 1$.
9. Return S.

Fig. 1. The enumeration algorithm

3.2 Solving SVP

To solve SVP, Kannan provides an algorithm that computes HKZ-reduced bases, see Figure 2. The cost of the enumeration procedure dominates the overall cost and mostly depends on the quality of the input basis. The main idea of Kannan's algorithm is to spend a lot of time pre-computing a basis of excellent quality before calling the enumeration procedure. More precisely, it pre-computes a so-called quasi-HKZ-reduced basis.

Definition 1 (Quasi-HKZ-reduction). *A basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ is quasi-HKZ-reduced if it is size-reduced, if $\|\boldsymbol{b}_2^*\| \geq \|\boldsymbol{b}_1^*\|/2$ and if once projected orthogonally to $\boldsymbol{b}_1$, the other $\boldsymbol{b}_i$'s are HKZ-reduced.*

A few comments need to be made on the algorithm of Figure 2. Steps 3 and 9 are recursive calls. However, the $\boldsymbol{b}_i'$'s may be rational vectors, whereas the input of the algorithm must be integral. These vectors may be scaled by a common factor. Steps 4 and 10 may be performed by expressing the reduced basis vectors as integer linear combinations of the initial ones, using these coefficients to recover lattice vectors and subtracting a correct multiple of the vector $\boldsymbol{b}_1$. In Step 6, it is possible to choose such a vector $\boldsymbol{b}_0$, since this enumeration always provides non-zero solutions (the vector $\boldsymbol{b}_1$ is one of them).

Input: An integer lattice basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$.
Output: An HKZ-reduced basis of the same lattice.
1. L^3-reduce the basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$.
2. Compute the projections $(\boldsymbol{b}'_i)_{i \geq 2}$ of the $\boldsymbol{b}_i$'s orthogonally to $\boldsymbol{b}_1$.
3. HKZ-reduce the $(d-1)$-dimensional basis $(\boldsymbol{b}'_2, \ldots, \boldsymbol{b}'_d)$.
4. Extend the obtained $(\boldsymbol{b}'_i)_{i \geq 2}$'s into vectors of L by adding to them rational multiples of $\boldsymbol{b}_1$, in such a way that we have $|\mu_{i,1}| \leq 1/2$ for any $i > 1$.
5. If $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ is not quasi-HKZ-reduced, swap $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ and go to Step 2.
6. Call the enumeration procedure to find all lattice vectors of length $\leq \|\boldsymbol{b}_1\|$. Let $\boldsymbol{b}_0$ be a shortest non-zero vector among them.
7. $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d) := \mathrm{L}^3(\boldsymbol{b}_0, \ldots, \boldsymbol{b}_d)$.
8. Compute the projections $(\boldsymbol{b}'_i)_{i \geq 2}$'s of the $\boldsymbol{b}_i$'s orthogonally to the vector $\boldsymbol{b}_1$.
9. HKZ-reduce the $(d-1)$-dimensional basis $(\boldsymbol{b}'_2, \ldots, \boldsymbol{b}'_d)$.
10. Extend the obtained $(\boldsymbol{b}'_i)_{i \geq 2}$'s into vectors of L by adding to them rational multiples of $\boldsymbol{b}_1$, in such a way that we have $|\mu_{i,1}| \leq 1/2$ for any $i > 1$.

Fig. 2. Kannan's SVP algorithm

3.3 Cost of Kannan's SVP Solver

We recall briefly Helfrich's analysis [13] of Kannan's algorithm and explain our complexity improvement. Let $C(d, n, B)$ be the worst-case complexity of the algorithm of Figure 2 when given as input a d-dimensional basis which is embedded in $\mathbb{Z}^n$ and whose coefficients are smaller than B in absolute value. The following properties hold:

- Kannan's algorithm computes an HKZ-reduced basis of the lattice spanned by the input vectors.
- All arithmetic operations performed during the execution are of cost $\mathcal{P}(d, n, \log B)$. This implies that $C(d, n, B)$ can be bounded by $C(d) \cdot \mathcal{P}(\log B, n)$ for some function $C(d)$.
- There are fewer than $O(1) + \log d$ iterations of the loop of Steps 2–5.
- The cost of the call to the enumeration procedure at Step 6 is bounded by $\mathcal{P}(\log B, n) \cdot d^{d/2+o(d)}$.

From these properties and those of the L^3 algorithm as recalled in the previous section, it is easy to obtain the following equation:

$$C(d) \leq (O(1) + \log d)(C(d-1) + \mathcal{P}(d)) + \mathcal{P}(d) + d^{\frac{d}{2}+o(d)}.$$

One can then derive the bound $C(d, B, n) \leq \mathcal{P}(\log B, n) \cdot d^{\frac{d}{2}+o(d)}$.

The main result of the present paper is to improve this complexity upper bound to $\mathcal{P}(\log B, n) \cdot d^{\frac{d}{2e}+o(d)}$. In fact, we show the following:

Theorem 2. *Given as inputs a quasi-HKZ-reduced basis* $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ *and* $A = \|\boldsymbol{b}_1\|^2$*, there are* $2^{O(d)} \cdot d^{\frac{d}{2e}}$ *loop iterations during the execution of the enumeration*

algorithm as described in Figure 1. As a consequence, given a d-dimensional basis of n-dimensional vectors whose entries are integers with absolute values $\leq B$, one can compute an HKZ-reduced basis of the spanned lattice in deterministic time $\mathcal{P}(\log B, n) \cdot d^{\frac{d}{2e}+o(d)}$.

4 Complexity of the Enumeration Procedure

This section is devoted to proving Theorem 2. The previous section has shown that the cost of Kannan's algorithm is dominated by the time for enumerating the integer points in the hyper-ellipsoids $(\mathcal{E}_i)_{1\leq i\leq d}$ defined by $\mathcal{E}_i = \left\{(y_i,\ldots,y_d) \in \mathbb{R}^{d-i+1}, \| \sum_{j\geq i} y_j \boldsymbol{b}_j^{(i)} \| \leq \|\boldsymbol{b}_1\|\right\}$, where $\boldsymbol{b}_j^{(i)} = \boldsymbol{b}_j - \sum_{k<i} \mu_{j,k}\boldsymbol{b}_k^*$ is the vector $\boldsymbol{b}_j$ once projected orthogonally to $\boldsymbol{b}_1^*,\ldots,\boldsymbol{b}_{i-1}^*$. Classically, the number of integer points in a body of some $\mathbb{R}^n$ is heuristically estimated by the n-dimensional volume of the body. This yields the following heuristic complexity upper-bound for Kannan's algorithm:

$$\max_{i\leq d} \frac{V_i \|\boldsymbol{b}_1\|^i}{\prod_{j\geq d-i+1} \|\boldsymbol{b}_j^*\|} \lesssim \max_{i\leq d} \frac{\|\boldsymbol{b}_1\|^i}{(\sqrt{i})^i \cdot \prod_{j\geq d-i+1} \|\boldsymbol{b}_j^*\|}, \tag{1}$$

where V_i is the volume of the i-dimensional unit ball.

Here, such an estimate may be too optimistic since the hyper-ellipsoids might be too flat for the approximation by the volume to be valid. The first step of our analysis is to prove a slight modification of this heuristic estimate. This is essentially an adaptation of a method due to Mazo and Odlyzko [20] to bound the number of integer points in hyper-spheres. We prove the weaker upper bound $\max_{I\subset[\![1,d]\!]} \frac{\|\boldsymbol{b}_1\|^{|I|}}{\sqrt{d}^{|I|}\prod_{i\in I}\|\boldsymbol{b}_i^*\|}$, for quasi-HKZ-reduced bases (Subsections 4.1 and 4.2).

In the second step of our analysis (Subsection 4.3), we bound the above quantity. This involves a rather precise study of the geometry of HKZ-reduced bases. The only available tool is Minkowski's inequality, which is used numerous times. For the intuition, the reader should consider the typical case where $(\boldsymbol{b}_i)_{1\leq i\leq d}$ is an HKZ-reduced basis for which $(\|\boldsymbol{b}_i^*\|)_i$ is a non-increasing sequence. In that case, the first part of the analysis shows that one has to consider a set I of much simpler shape: it is an interval $[\![i,d]\!]$ starting at some index i. Lemmata 2 and 3 (which should thus be considered as the core of the proof) and the fact that $x\log x \geq -1/e$ for $x\in[0,1]$ are sufficient to deal with such sets.

Non-connex sets I are harder to handle. We split the HKZ-reduced basis into *blocks* (defined by the expression of I as a union of intervals), i.e., groups of consecutive vectors $\boldsymbol{b}_i,\ldots,\boldsymbol{b}_{j-1}$ such that $i,\ldots,k-1\notin I$ and $k,\ldots,j-1\in I$. The former vectors will be the "large ones" and the latter the "small ones". Over each block, Lemma 3 relates the average size of the small vectors to the average size of the whole block. We consider the blocks by decreasing indices and use an amortised analysis to combine the local behaviours on blocks to obtain a global bound (Lemma 4). A final convexity argument gives the result (Lemma 5).

4.1 Integer Points in Hyper-Ellipsoids

In this subsection, we do not assume anything on the input basis vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d$ and on the input bound A. Up to some polynomial in d and $\log B$, the complexity of the enumeration procedure of Figure 1 is the number of loop iterations. This number of iterations is itself bounded by $3\sum_{i=1}^{d} |\mathcal{E}_i|$. Indeed, the truncated coordinate $(x_i, \ldots, x_d)$ is either a valid one, i.e., we have $\|\sum_{j=i}^{d} x_j \boldsymbol{b}_j^{(i)}\|^2 \leq A$, or $(x_i - 1, \ldots, x_d)$ is a valid one, or $(x_{i+1}, \ldots, x_d)$ is a valid one. In fact, if $(x_i, \ldots, x_d)$ is a valid truncated coordinate, at most two non-valid ones related to that one may be considered during the execution of the algorithm: $(x_i + 1, \ldots, x_d)$ and $(x_{i-1}, x_i \ldots, x_d)$ for at most one integer x_{i-1}. We now fix some $i \leq d$. By applying the change of variable $x_j \leftarrow x_j - \left\lfloor \sum_{k>j} \mu_{k,j} x_k \right\rceil$, we obtain:

$$|\mathcal{E}_{d-i+1}| \leq \left| \left\{ (x_j)_{i \leq j \leq d} \in \mathbb{Z}^{d-i+1}, \sum_{j \geq i} (x_j + \sum_{k>j} \mu_{k,j} x_k)^2 \cdot \|\boldsymbol{b}_j^*\|^2 \leq A \right\} \right|$$
$$\leq \left| \left\{ (x_j)_{i \leq j \leq d} \in \mathbb{Z}^{d-i+1}, \sum_{j \geq i} (x_j + \{\sum_{k>j} \mu_{k,j} x_k\})^2 \cdot \|\boldsymbol{b}_j^*\|^2 \leq A \right\} \right|.$$

If x is an integer and $\epsilon \in [-1/2, 1/2]$, then we have $(x + \epsilon)^2 \geq x^2/4$ (it suffices to use the inequality $|\epsilon| \leq 1/2 \leq |x|/2$, which is valid for a non-zero x). As a consequence, up to a polynomial factor, the complexity of the enumeration is bounded by $\sum_{i \leq d} N_i$, where $N_i = \left|\mathcal{E}_i' \cap \mathbb{Z}^{d-i+1}\right|$ and $\mathcal{E}_i' = \left\{ (y_i, \ldots, y_d) \in \mathbb{R}^{d-i+1}, \sum_{j \geq i} y_j^2 \|\boldsymbol{b}_j^*\|^2 \leq 4A \right\}$, for any $i \leq d$.

We again fix some index i. The following sequence of relations is inspired from [20, Lemma 1].

$$N_i = \sum_{(x_i, \ldots, x_d) \in \mathbb{Z}^{d-i+1}} \mathbf{1}_{\mathcal{E}_i'}(x_i, \ldots, x_d) \leq \exp\left(d \left(1 - \sum_{j \geq i} x_j^2 \frac{\|\boldsymbol{b}_j^*\|^2}{4A} \right) \right)$$
$$\leq e^d \cdot \prod_{j \geq i} \sum_{x \in \mathbb{Z}} \exp\left(-x^2 \frac{d\|\boldsymbol{b}_j^*\|^2}{4A} \right) = e^d \cdot \prod_{j \geq i} \Theta\left(\frac{d\|\boldsymbol{b}_j^*\|^2}{4A} \right),$$

where $\Theta(t) = \sum_{x \in \mathbb{Z}} \exp(-tx^2)$ is defined for $t > 0$. Notice that $\Theta(t) = 1 + 2\sum_{x \geq 1} \exp(-tx^2) \leq 1 + 2\int_0^\infty \exp(-tx^2) dx = 1 + \sqrt{\frac{\pi}{t}}$. Hence $\Theta(t) \leq \frac{1+\sqrt{\pi}}{\sqrt{t}}$ for $t \leq 1$ and $\Theta(t) \leq 1 + \sqrt{\pi}$ for $t \geq 1$. As a consequence, we have:

$$N_i \leq (4e(1 + \sqrt{\pi}))^d \cdot \prod_{j \geq i} \max\left(1, \frac{\sqrt{A}}{\sqrt{d}\|\boldsymbol{b}_j^*\|} \right). \tag{2}$$

One thus concludes that the cost of the enumeration is bounded by:

$$\mathcal{P}(n, \log A, \log B) \cdot 2^{O(d)} \cdot \max_{I \subset [\![1,d]\!]} \left(\frac{(\sqrt{A})^{|I|}}{(\sqrt{d})^{|I|} \prod_{i \in I} \|\boldsymbol{b}_i^*\|} \right).$$

4.2 The Case of Quasi-HKZ-Reduced Bases

We now suppose that $A = \|\boldsymbol{b}_1\|^2$ and that the input basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ is quasi-HKZ-reduced. We are to strengthen the quasi-HKZ-reducedness hypothesis into an HKZ-reducedness hypothesis. Let $I \subset [\![1, d]\!]$. If $1 \notin I$, then, because of the quasi-HKZ-reducedness assumption:

$$\frac{\|\boldsymbol{b}_1\|^{|I|}}{(\sqrt{d})^{|I|}\prod_{i\in I}\|\boldsymbol{b}_i^*\|} \leq 2^d \frac{\|\boldsymbol{b}_2^*\|^{|I|}}{(\sqrt{d})^{|I|}\prod_{i\in I}\|\boldsymbol{b}_i^*\|}.$$

If $1 \in I$, we have, by removing $\|\boldsymbol{b}_1^*\|$ from the product $\prod_{i\in I-\{1\}}\|\boldsymbol{b}_i^*\|$:

$$\frac{\|\boldsymbol{b}_1\|^{|I|}}{(\sqrt{d})^{|I|}\prod_{i\in I}\|\boldsymbol{b}_i^*\|} \leq 2^d \frac{\|\boldsymbol{b}_2^*\|^{|I|-1}}{(\sqrt{d})^{|I|-1}\prod_{i\in I-\{1\}}\|\boldsymbol{b}_i^*\|}.$$

As a consequence, Theorem 2 follows from the following:

Theorem 3. *Let $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ be HKZ-reduced and $I \subset [\![1, d]\!]$. Then*

$$\frac{\|\boldsymbol{b}_1\|^{|I|}}{\prod_{i\in I}\|\boldsymbol{b}_i^*\|} \leq (\sqrt{d})^{|I|\left(1+\log\frac{d}{|I|}\right)} \leq (\sqrt{d})^{\frac{d}{e}+|I|}.$$

By applying Theorem 3 the HKZ-reduced basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_i)$ and $I = \{i\}$, we recover the result of [28]: $\|\boldsymbol{b}_i^*\| \geq (\sqrt{i})^{-\log i - 1} \cdot \|\boldsymbol{b}_1\|$.

4.3 A Property on the Geometry of HKZ-Reduced Bases

In this section, we prove Theorem 3, which is the last missing part to obtain the claimed result. The proofs of the following lemmata will be contained in the full version of this paper. In the sequel, $(\boldsymbol{b}_i)_{i\leq d}$ is an HKZ-reduced basis of a lattice L of dimension $d \geq 2$.

Definition 2. *For any $I \subset [\![1, d]\!]$, we define $\pi_I = \left(\prod_{i\in I}\|\boldsymbol{b}_i^*\|\right)^{\frac{1}{|I|}}$. Moreover, if $k \in [\![1, d-1]\!]$, we define $\Gamma_d(k) = \prod_{i=d-k}^{d-1}(\gamma_{i+1})^{\frac{1}{2i}}$.*

We need upper bounds on $\Gamma_d(k)$ and a technical lemma allowing us to finely recombine such bounds. Intuitively, the following lemma is a rigorous version of the identity:

$$\log\Gamma_d(k) \approx \int_{x=d-k}^{d}\frac{1}{2x}\log x\,\mathrm{d}x \approx \frac{\log^2(d) - \log^2(d-k)}{4} \lesssim \frac{\log d}{2}\log\frac{d}{d-k}.$$

Lemma 2. *For all $1 \leq k < d$, we have $\Gamma_d(k) \leq \sqrt{d}^{\log\frac{d}{d-k}}$.*

We now give an "averaged" version of [28, Lemma 4], deriving from Lemma 2. This provides the result claimed in Theorem 3 for any set I of the shape $[\![i, j]\!]$, for any $i \leq j \leq d$.

Lemma 3. *For all $k \in [\![0, d-1]\!]$, we have $\pi_{[\![1,k]\!]} \leq (\Gamma_d(k))^{d/k} \cdot \pi_{[\![k+1,d]\!]}$ and $\pi_{[\![k+1,d]\!]} \geq (\Gamma_d(k))^{-1} \cdot (\det L)^{1/d} \geq \sqrt{d}^{\log \frac{d-k}{d}} (\det L)^{1/d}$.*

We prove Theorem 3 by induction on the number of intervals occurring in the expression of the set I as a union of intervals. The following lemma is the induction step. This is a recombination step, where we join one block (between the indices 1 and v, the "small vectors" being those between $u+1$ and v) to one or more already considered blocks on its right. An important point is to ensure that the densities δ_i defined below actually decrease when their indices increase. Its proof is based on Lemma 3.

Lemma 4. *Let $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ be an HKZ-reduced basis. Let $v \in [\![2, d]\!]$, $I \subset [\![v+1, d]\!]$ and $u \in [\![1, v]\!]$. Assume that:*

$$\pi_I^{|I|} \geq \prod_{i<t} \left(\pi_{[\![\alpha_i+1, \alpha_{i+1}]\!]}^{|I_i|} \cdot \sqrt{d}^{|I_i| \log \delta_i} \right),$$

where $I_i = I \cap [\![\alpha_i+1, \alpha_{i+1}]\!]$, $\delta_i = \frac{|I_i|}{\alpha_{i+1}-\alpha_i}$ is the density of the set I in $[\![\alpha_i+1, \alpha_{i+1}]\!]$, and the integers t and α_i's, and the densities δ_i's satisfy $t \geq 1$, $v = \alpha_1 < \ldots < \alpha_t \leq d$ and $1 \geq \delta_1 > \ldots > \delta_{t-1} > 0$. Then, we have

$$\pi_{I'}^{|I'|} \geq \prod_{i<t'} \left(\pi_{[\![\alpha'_i+1, \alpha'_{i+1}]\!]}^{|I'_i|} \cdot \sqrt{d}^{|I'_i| \log \delta'_i} \right),$$

where $I' = [\![u+1, v]\!] \cup I$, $I'_i = I' \cap [\![\alpha'_i+1, \alpha'_{i+1}]\!]$, $\delta'_i = \frac{|I'_i|}{\alpha'_{i+1}-\alpha'_i}$ and the integers t' and α'_i's, and the densities δ'_i satisfy $t' \geq 1$, $0 = \alpha'_1 < \ldots < \alpha'_{t'} \leq d$ and $1 \geq \delta'_1 > \ldots > \delta'_{t'-1} > 0$.

The last ingredient to the proof of Theorem 3 is the following, which derives from the convexity of the function $x \mapsto x \log x$.

Lemma 5. *Let $\Delta \geq 1$, and define $F_\Delta(k, d) = \Delta^{-k \log \frac{k}{d}}$. We have, for any $t \in \mathbb{Z}$, for any $k_1, \ldots, k_t \in \mathbb{Z}$ and $d_1, \ldots, d_t \in \mathbb{Z}$ such that $1 \leq k_i < d_i$ for all $i \leq t$,*

$$\prod_{i \leq t} F_\Delta(k_i, d_i) \leq F_\Delta \left(\sum_{i \leq t} k_i, \sum_{i \leq t} d_i \right).$$

Finally, Theorem 3 follows from Lemmata 4 and 5.

Proof of Theorem 3. Lemma 4 gives us, by induction on the size of the considered set I, that for all $I \subset [\![1, d]\!]$:

$$\pi_I^{|I|} \geq \prod_{i<t} \left(\pi_{[\![\alpha_i+1, \alpha_{i+1}]\!]}^{|I_i|} \cdot \sqrt{d}^{|I_i| \log \delta_i} \right),$$

where $I_i = I \cap [\![\alpha_i + 1, \alpha_{i+1}]\!]$, and t, the α_i's, and the densities $\delta_i = \frac{|I_i|}{\alpha_{i+1}-\alpha_i}$ satisfy $t \geq 1$, $0 = \alpha_1 < \ldots < \alpha_t \leq d$ and $1 \geq \delta_1 > \ldots > \delta_{t-1} > 0$. By using

Lemma 5 with $\Delta{:=}\sqrt{d}, k_i{:=}|I_i|$ and $d_i{:=}\alpha_{i+1}-\alpha_i$, we obtain:

$$\pi_I^{|I|} \geq \left(\sqrt{d}^{|I|\log\frac{|I|}{\alpha_t-\alpha_1}}\right)\cdot\left(\prod_{i<t}\pi_{[\![\alpha_i+1,\alpha_{i+1}]\!]}^{|I_i|}\right).$$

We define $\delta_t = 0$. Because of the definition of the α_i's, we have:

$$\prod_{i<t}\pi_{[\![\alpha_i+1,\alpha_{i+1}]\!]}^{|I_i|} = \prod_{i<t}\left(\pi_{[\![\alpha_i+1,\alpha_{i+1}]\!]}^{\alpha_{i+1}-\alpha_i}\right)^{\delta_i} = \prod_{i<t}\prod_{i\leq j<t}\left(\pi_{[\![\alpha_i+1,\alpha_{i+1}]\!]}^{\alpha_{i+1}-\alpha_i}\right)^{\delta_j-\delta_{j+1}}$$
$$= \prod_{j<t}\left(\prod_{i\leq j}\pi_{[\![\alpha_i+1,\alpha_{i+1}]\!]}^{\alpha_{i+1}-\alpha_i}\right)^{\delta_j-\delta_{j+1}} = \prod_{j<t}\left(\pi_{[\![1,\alpha_{j+1}]\!]}^{\alpha_{j+1}}\right)^{\delta_j-\delta_{j+1}}.$$

By using $t-1$ times Minkowski's theorem, we obtain that:

$$\frac{\pi_I^{|I|}}{\sqrt{d}^{|I|\log\frac{|I|}{d}}} \geq \left(\frac{\|\boldsymbol{b}_1\|}{\sqrt{d}}\right)^{\sum_{j<t}\alpha_{j+1}(\delta_j-\delta_{j+1})} \geq \left(\frac{\|\boldsymbol{b}_1\|}{\sqrt{d}}\right)^{|I|}.$$

The final inequality of the theorem comes from the fact that the function $x \mapsto x\log(d/x)$ is maximal for $x = d/e$. □

5 CVP and Other Related Problems

Our improved analysis of Kannan's algorithm can be adapted to the Closest Vector Problem and other problems related to strong lattice reduction.

In CVP, we are given a basis $(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d)$ and a target vector $\boldsymbol{t}$, and we look for a lattice vector that is closest to $\boldsymbol{t}$. Kannan's CVP algorithm starts by HKZ-reducing the $\boldsymbol{b}_i$'s. Then it runs a slight modification of the enumeration algorithm of Figure 1. For the sake of simplicity, we assume that $\|\boldsymbol{b}_1^*\|$ is the largest of the $\|\boldsymbol{b}_i^*\|$'s (we refer to Kannan's proof [16] for the general case). By using Babai's nearest hyperplane strategy [6], we see that there is a lattice vector $\boldsymbol{b}$ at distance less than $\sqrt{d}\cdot\|\boldsymbol{b}_1\|$ of the target vector $\boldsymbol{t}$. As a consequence, if we take $A = d\cdot\|\boldsymbol{b}_1\|^2$ in the modified enumeration procedure, we will find all solutions. The analysis then reduces (at the level of Equation (2)) to bound the ratio $\frac{\|\boldsymbol{b}_1\|^d}{\prod_{i\leq d}\|\boldsymbol{b}_i^*\|}$, which can be done with Minkowski's theorem.

Theorem 4. *Given a basis $(\boldsymbol{b}_1,\ldots,\boldsymbol{b}_d)$ and a target vector $\boldsymbol{t}$, all of them in $\mathbb{Z}^n$ and with coordinates whose absolute values are smaller than some B, one can compute all vectors in the lattice spanned by the $\boldsymbol{b}_i$'s that are closest to $\boldsymbol{t}$ in deterministic time $\mathcal{P}(\log B, n)\cdot d^{d/2+o(d)}$.*

The best deterministic complexity upper bound previously known for this problem was $\mathcal{P}(\log B, n)\cdot d^{d+o(d)}$ (see [13, 7]).

Our result can also be adapted to the enumeration of all vectors of a given lattice that are of length below a prescribed bound, which is in particular useful in the context of computing lattice theta series. Another important consequence of our analysis is a significant worst-case bound improvement of Schnorr's

block-based strategy [28] to compute relatively short vectors in high-dimensional lattices. More precisely, if we take the bounds given in [10] for the quality of Schnorr's semi-$2k$ reduction and for the transference reduction, we obtain the table of Figure 3. Each entry of the table gives the upper bound of the quantity $\frac{\|\boldsymbol{b}_1\|}{(\det L)^{1/d}}$ which is reachable for a computational effort of 2^t, for t growing to infinity. To sum up, the exponent constant is divided by $e \approx 2.7$. The table upper bounds may be adapted to the quantity $\frac{\|\boldsymbol{b}_1\|}{\lambda_1(L)}$ by squaring them.

	Semi-$2k$ reduction	Transference reduction
Using [13]	$\lesssim 2^{\frac{\log 2}{2}\frac{d\log^2 t}{t}} \approx 2^{0.347\frac{d\log^2 t}{t}}$	$\lesssim 2^{\frac{1}{4}\frac{d\log^2 t}{t}} \approx 2^{0.250\frac{d\log^2 t}{t}}$
Using Theorem 2	$\lesssim 2^{\frac{\log 2}{2e}\frac{d\log^2 t}{t}} \approx 2^{0.128\frac{d\log^2 t}{t}}$	$\lesssim 2^{\frac{1}{4e}\frac{d\log^2 t}{t}} \approx 2^{0.092\frac{d\log^2 t}{t}}$

Fig. 3. Worst-case bounds for block-based reduction algorithms

6 Practical Implications

As mentioned in the introduction, the main contribution of the present paper is to improve the worst-case complexity analysis of an already known algorithm, namely, Kannan's HKZ-reduction algorithm. Our improvement has no direct impact on the practical capabilities of lattice reduction algorithms. However, our work may have two indirect consequences: popularising Kannan's principle and providing easily computable cost estimates for SVP instances.

6.1 Pre-processing Before Enumerating

In the main libraries containing lattice reduction routines, the shortest vector problem is solved with the enumeration routine, but starting from only L^3-reduced bases. This is the case for the `BKZ` routines of Victor Shoup's NTL [32], which, depending on a parameter k, compute strongly reduced bases in high dimensions (the quality being quantified by k). This is also the case in Magma's `ShortestVectors` routine [18], which computes the shortest vectors of a given lattice. Both rely on the enumeration of Schnorr and Euchner [30]. On the theoretical side, this strategy is worse than using Kannan's algorithm, the worst-case complexity being $2^{O(d^2)}$ instead of $d^{O(d)}$. To justify this choice, one might argue that L^3 computes much better bases in practice than guaranteed by the worst-case bounds, in particular in low dimensions (see [24] for more details), and that the asymptotically superior algorithm of Kannan may overtake the L^3-based enumeration only for large dimensions (in particular too large to be tractable).

It may be that the genuine Kannan algorithm is expensive. However, the general principle of enumerating from a more than L^3-reduced basis works, as the following experiments tend to show. For a given dimension d, we consider the lattice spanned by the columns of the following matrix:

$$\begin{pmatrix} x_1 & x_2 & \dots & x_d \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix},$$

where the x_i's are chosen uniformly and independently in $[\![0, 2^{100 \cdot d}]\!]$. The basis is then L^3-reduced with a close to optimal parameter ($\delta = 0.99$). For the same lattice, we compute more reduced bases, namely BKZ_k-reduced for different parameters k, using NTL's `BKZ_FP` routine without pruning and close to optimal factor ($\delta = 0.99$). We run the same enumeration routine starting from these different bases and compare the timings. The results of the experiments are given in Figure 4. The enumeration is a non-optimised C-code, which updates the norm upper bound during the enumeration [30]. All timings are given in seconds and include the BKZ-reduction (unless we start from the L^3-reduced basis). Each point corresponds to the average over at least 10 samples. The experiments were performed on 2.4 GHz AMD Opterons. The enumeration from an L^3-reduced basis is clearly outperformed. BKZ-reducing the basis with larger block-sizes becomes more interesting when the dimension increases: it seems that in moderate dimension, a BKZ_k reduced basis is close to being HKZ-reduced, even when k is small with respect to the dimension.

pre-processing	$d = 40$	$d = 43$	$d = 46$	$d = 49$	$d = 52$	$d = 55$	$d = 58$
L^3	1.8	15	110	990	$5.0 \cdot 10^3$	–	–
BKZ_{10}	0.36	1.6	6.7	36	160	–	–
BKZ_{20}	0.40	1.3	4.7	21	96	800	$2.5 \cdot 10^3$
BKZ_{30}	0.57	1.7	5.2	19	68	660	$1.6 \cdot 10^3$

Fig. 4. Comparison between various pre-processings

6.2 Estimating the Cost of Solving SVP

The cost of solving SVP on a particular instance with the enumeration routine is essentially dominated by the cost of the highest-dimensional enumeration. Up to a polynomial factor, the cost of the enumeration as described in Figure 1 can be estimated with Equation (1):

$$E(\boldsymbol{b}_1, \dots, \boldsymbol{b}_d) := \max_{i \leq d} \frac{\pi^{i/2} \cdot \|\boldsymbol{b}_1\|^i}{\Gamma(i/2+1) \cdot \prod_{j \geq d-i+1} \|\boldsymbol{b}_j^*\|}.$$

This estimate is simply the application of the Gaussian heuristic, stating that the number of integer points within a body is essentially the volume of the body. It can be computed in polynomial time from the basis from which the enumeration will be started. We computed $E(\boldsymbol{b}_1, \dots, \boldsymbol{b}_d)$ for random bases generated as above

pre-processing	$d = 40$	$d = 45$	$d = 50$	$d = 55$	$d = 60$	$d = 65$	$d = 70$	$d = 75$
L^3	$1.0 \cdot 10^8$	$4.4 \cdot 10^9$	$1.5 \cdot 10^{14}$	$9.6 \cdot 10^{16}$	$3.0 \cdot 10^{18}$	$6.1 \cdot 10^{21}$	$2.8 \cdot 10^{27}$	$1.6 \cdot 10^{30}$
BKZ_{10}	$4.6 \cdot 10^5$	$1.2 \cdot 10^7$	$1.1 \cdot 10^8$	$1.3 \cdot 10^{10}$	$7.6 \cdot 10^{11}$	$1.7 \cdot 10^{14}$	$4.3 \cdot 10^{16}$	$1.9 \cdot 10^{19}$
BKZ_{20}	$2.4 \cdot 10^5$	$2.7 \cdot 10^6$	$3.1 \cdot 10^7$	$1.3 \cdot 10^9$	$4.1 \cdot 10^{10}$	$3.7 \cdot 10^{12}$	$6.4 \cdot 10^{13}$	$2.1 \cdot 10^{16}$
BKZ_{30}	$1.9 \cdot 10^5$	$1.6 \cdot 10^6$	$1.8 \cdot 10^7$	$3.0 \cdot 10^8$	$4.3 \cdot 10^9$	$1.1 \cdot 10^{11}$	$3.7 \cdot 10^{12}$	$1.9 \cdot 10^{14}$

Fig. 5. Value of $E(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ for randomly generated $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$

and obtained the table of Figure 5. It confirms that a strong pre-processing should help increasing the dimension up to which SVP may be solved completely.

If one is looking for vectors smaller than some prescribed B (for example if the existence of an unusually short vector is promised), then $\|\boldsymbol{b}_1\|$ may be replaced by B in the estimate. Overall, these estimates are rather crude since factors that are polynomial in the dimension should be considered as well. Furthermore, it does not take into account more elaborate techniques such as updating the norm during the enumeration, pruning [30, 31] and random sampling [29].

Open problem. One may wonder if the complexity upper bound for Kannan's SVP algorithm can be decreased further. Work under progress seems to show, by using a technique due to Ajtai [3], that it is sharp, in the sense that for all $\epsilon > 0$, we can build HKZ-reduced bases for which the number of steps of Kannan's algorithm would be at least $d^{d\left(\frac{1}{2e} - \epsilon\right)}$.

Acknowledgements. We thank Frederik Vercauteren for helpful discussions, as well as John Cannon and the University of Sydney for having hosted the second author while a large part of this work was completed.

References

1. Agrell, E., Eriksson, T., Vardy, A., Zeger, K.: Closest point search in lattices. IEEE Trans. Inform. Theory 48(8), 2201–2214 (2002)
2. Ajtai, M.: The shortest vector problem in l_2 is NP-hard for randomized reductions (extended abstract). In: Proc. of STOC 1998, pp. 284–293. ACM Press, New York (1998)
3. Ajtai, M.: The worst-case behavior of Schnorr's algorithm approximating the shortest nonzero vector in a lattice. In: Proc. of STOC 2003, pp. 396–406. ACM Press, New York (2003)
4. Ajtai, M., Dwork, C.: A public-key cryptosystem with worst-case/average-case equivalence. In: Proc. of STOC 1997, pp. 284–293. ACM Press, New York (1997)
5. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. In: Proc of STOC 2001, pp. 601–610. ACM Press, New York (2001)
6. Babai, L.: On Lovász lattice reduction and the nearest lattice point problem. Combinatorica 6, 1–13 (1986)
7. Blömer, J.: Closest vectors, successive minima and dual-HKZ bases of lattices. In: Welzl, E., Montanari, U., Rolim, J.D.P. (eds.) ICALP 2000. LNCS, vol. 1853, pp. 248–259. Springer, Heidelberg (2000)

8. Coppersmith, D., Shamir, A.: Lattice attacks on NTRU. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 52–61. Springer, Heidelberg (1997)
9. Fincke, U., Pohst, M.: A procedure for determining algebraic integers of given norm. In: van Hulzen, J.A. (ed.) ISSAC 1983 and EUROCAL 1983. LNCS, vol. 162, pp. 194–202. Springer, Heidelberg (1983)
10. Gama, N., Howgrave-Graham, N., Koy, H., Nguyen, P.: Rankin's constant and blockwise lattice reduction. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 112–130. Springer, Heidelberg (2006)
11. Goldreich, O., Goldwasser, S., Halevi, S.: Public-key cryptosystems from lattice reduction problems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 112–131. Springer, Heidelberg (1997)
12. Haviv, I., Regev, O.: Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In: Proc. of STOC 2007 (2007)
13. Helfrich, B.: Algorithms to construct Minkowski reduced and Hermite reduced lattice bases. Theoret. Comput. Sci. 41, 125–139 (1985)
14. Hermite, C.: Extraits de lettres de M. Hermite à M. Jacobi sur différents objets de la théorie des nombres, deuxième lettre. J. Reine Angew. Math. 40, 279–290 (1850)
15. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU : a ring based public key cryptosystem. In: Buhler, J.P. (ed.) Algorithmic Number Theory. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998)
16. Kannan, R.: Improved algorithms for integer programming and related lattice problems. In: Proc. of STOC 1983, pp. 99–108. ACM Press, New York (1983)
17. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Math. Ann. 261, 513–534 (1982)
18. Magma. The Magma computational algebra system for algebra, number theory and geometry. Available at `http://magma.maths.usyd.edu.au/magma/`
19. Martinet, J.: Perfect Lattices in Euclidean Spaces. Springer, Heidelberg (2002)
20. Mazo, J., Odlyzko, A.: Lattice points in high-dimensional spheres. Monatsh. Math. 110, 47–61 (1990)
21. Micciancio, D., Goldwasser, S.: Complexity of lattice problems : a cryptographic perspective. Kluwer Academic Publishers, Dordrecht (2002)
22. Minkowski, H.: Geometrie der Zahlen. Teubner-V (1896)
23. Nguyen, P.: Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto'97. In: Wiener, M.J. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 288–304. Springer, Heidelberg (1999)
24. Nguyen, P., Stehlé, D.: LLL on the average. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTSVII. LNCS, vol. 4076, pp. 238–256. Springer, Heidelberg (2006)
25. Nguyen, P., Stern, J.: Cryptanalysis of the Ajtai-Dwork cryptosystem. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 223–242. Springer, Heidelberg (1998)
26. Nguyen, P., Vidick, T.: Assessing sieve algorithms for the shortest vector problem. Draft (2007)
27. Regev, O.: Lecture notes of lattices in computer science, taught at the Computer Science Tel Aviv University. Available at `http://www.cs.tau.il/~odedr`
28. Schnorr, C.P.: A hierarchy of polynomial lattice basis reduction algorithms. Theoret. Comput. Sci. 53, 201–224 (1987)

29. Schnorr, C.P.: Lattice reduction by random sampling and birthday methods. In: Alt, H., Habib, M. (eds.) STACS 2003. LNCS, vol. 2607, pp. 145–156. Springer, Heidelberg (2003)
30. Schnorr, C.P., Euchner, M.: Lattice basis reduction : improved practical algorithms and solving subset sum problems. Math. Programming 66, 181–199 (1994)
31. 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)
32. Shoup, V.: NTL, Number Theory Library. Available, at`http://www.shoup.net/ntl/`

Analyzing Blockwise Lattice Algorithms using Dynamical Systems

Guillaume Hanrot, Xavier Pujol, and Damien Stehlé

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 $\lambda_1(L)$ 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 [40,9].

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 [10,14]. 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 [33,37,34,7,32,9,8]). From a theoretical perspective, the best known time/quality trade-off is due to Gama and Nguyen [8]. 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 $\leq \mathcal{P}oly(n, \text{size}(B)) \cdot \mathcal{C}_{\texttt{HKZ}}(\beta)$, 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 $\text{vol}(\mathbb{R}^n/L)$):

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

From a practical perspective, however, slide reduction seems to be (significantly) outperformed by the BKZ algorithm [9]. 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 [9]. 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 [9] 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* $\tau_{\texttt{BKZ}} := C\frac{n^3}{\beta^2}\left(\log n + \log\log \max_i \frac{\|\boldsymbol{b}_i\|}{(\det L)^{1/n}}\right)$ *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* $\leq \mathcal{P}oly(n, \text{size}(B)) \cdot \mathcal{C}_{\texttt{HKZ}}(\beta)$*.*

By using [18, p. 25], this provides an algorithm with runtime bounded by $\mathcal{P}oly(n, \text{size}(B)) \cdot \mathcal{C}_{\texttt{HKZ}}(\beta)$ 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 $\tau_{\texttt{BKZ}}$ is significantly smaller than $\tau_{\texttt{slide}}$, in particular with a dependence with

[1] The component $\frac{n^4}{\beta}$ of this upper bound is derived by adapting the results from [8] to our notations. A more thorough analysis leads to a smaller term.

[2] In [9], 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 appendix.

[3] A bound $(n\beta)^n$ is mentioned in [9]. For completeness, we give a proof of a similar result in appendix.

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

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.

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 analyzes of (variants of) LLL we are aware of. It was later adapted to stronger lattice reduction algorithms [33,7,32,8]. 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 some 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 $2^{2n+o(n)} \cdot \mathcal{P}oly(\mathrm{size}(B))$, 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 $\simeq 1.012$.

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

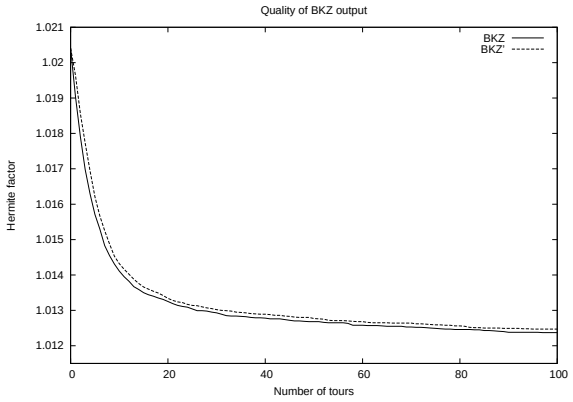

Fig. 1. Evolution of the Hermite factor $\frac{\|\boldsymbol{b}_1\|}{(\det L)^{1/n}}$ during the execution of BKZ and BKZ'.

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 analyzes 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* $0 \leq k < n$.

Proof. SMA is equivalent to the following triangular system of linear equations:

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

Let $y_i = \sum_{j=i}^{\beta} x_j$, for $i \leq \beta$. Then $y_\beta = x_\beta$ and $y_i = \frac{\beta - i + 1}{\beta - i}\left(y_{i+1} + \frac{1}{2}\log \nu_{\beta - i + 1}\right)$ for all $i < \beta$. By induction:

$$\forall i \leq \beta,\ y_i = (\beta - i + 1)\left(y_\beta + \sum_{j=1}^{\beta - i} \frac{\log \nu_{j+1}}{2j}\right).$$

Taking $i = 1$ and noting that $y_1 = \beta \cdot \mathbb{E}[\boldsymbol{x}]$ gives $y_\beta = x_\beta = \mathbb{E}[\boldsymbol{x}] - \Gamma_\beta(\beta - 1)$. Now:

$$\forall i < \beta,\ y_i = (\beta - i + 1)\left(\mathbb{E}[\boldsymbol{x}] - \Gamma_\beta(\beta - 1) + \sum_{j=1}^{\beta - i} \frac{\log \nu_{j+1}}{2j}\right) = (\beta - i + 1)\left(\mathbb{E}[\boldsymbol{x}] - \Gamma_\beta(i - 1)\right).$$

The result derives from the equality $x_i = y_i - y_{i+1}$. □

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:

$$A^{(\alpha)} = \begin{pmatrix} \ddots & & & & & \\ & 1 & & & & \\ & & \frac{1}{\beta} \cdots \frac{1}{\beta} & & & \\ & & \vdots \ \ddots \ \vdots & & & \\ & & \frac{1}{\beta} \cdots \frac{1}{\beta} & & & \\ & & & 1 & & \\ & & & & \ddots & \end{pmatrix} \begin{matrix} \\ \\ (\alpha) \\ \\ (\alpha+\beta-1) \\ \\ \end{matrix} \quad \text{and } g_i^{(\alpha)} = \begin{cases} 0 \ \text{ if } \ i < \alpha \\ (\beta + \alpha - i - 1)\Gamma_\beta(i - \alpha + 1) - (\beta + \alpha - i)\Gamma_\beta(i - \alpha) \\ \qquad\qquad\qquad\qquad\qquad\qquad \text{if } i \in [\alpha, \alpha + \beta - 2] \\ -\Gamma_\beta(\beta - 1) \ \text{ if } \ i = \alpha + \beta - 1 \\ 0 \ \text{ if } \ i \geq \alpha + \beta. \end{cases}$$

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 $(\boldsymbol{b}_i)_{i \le n}$ 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 $(\boldsymbol{c}_i)_{i \le n}$ is a basis of L that satisfies $\|\boldsymbol{x} - \boldsymbol{x}^\infty\|_2 \le 1$, 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]

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

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 $\boldsymbol{x} \in \mathrm{span}(1, \ldots, 1)^T$.*

Proof. Let $\boldsymbol{x} \in \mathbb{R}^n$ such that $A \cdot \boldsymbol{x} = \boldsymbol{x}$. Let i the largest index such that $x_i = \max_j x_j$. We prove by contradiction that $i = n$. Assume that $i < n$. We consider two cases, depending on whether $i < \beta$ or $i \ge \beta$. Recall that applying $A^{(\alpha)}$ to a vector $\boldsymbol{y}$ consists in replacing $y_\alpha, \ldots, y_{\alpha+\beta-1}$ by their mean, and in leaving the others constant. As a result, the maximum of the y_i's cannot increase.

Assume first that $i < \beta$. Let $\boldsymbol{x}' = A^{(1)} \cdot \boldsymbol{x}$. By definition of i, we must have $x_{i+1} < x_i$, and therefore $\max_{j \le \beta} x'_j < \max_{j \le \beta} x_j$. By choice of i, we also have $\max_{j \le n} x'_j < \max_{j \le n} x_j$. But $\boldsymbol{x} = A^{(n-\beta+1)} \cdot \ldots \cdot A^{(2)}\boldsymbol{x}'$, which leads to the inequality $\max_{j \le n} x_j \le \max_{j \le n} x'_j$. We obtained a contradiction.

Now, assume that $i \ge \beta$. Let $\boldsymbol{x}' = A^{(i-\beta+1)} \cdot \ldots \cdot A^{(1)} \cdot \boldsymbol{x}$ and $\boldsymbol{x}'' = A^{(i-\beta+2)} \cdot \boldsymbol{x}'$. We have $\max_{j \le n} x'_j \le \max_{j \le n} x_j = x_i$. Moreover, we have $x'_{i-\beta+1} = \ldots = x'_i \le x_i$ and for all $j > i$, $x'_j = x_j < x_i$. This implies that $\max_{i-\beta+2 \le j \le n} x''_j < x_i$. Since $\boldsymbol{x} = A^{(n-\beta+1)} \cdot \ldots \cdot A^{(i-\beta+3)} \cdot \boldsymbol{x}$", we obtain that $\max_{i-\beta+2 \le j \le n} x_j < x_i$. In particular, we obtain the contradiction $x_i < x_i$.

So far, we have proven that $x_n = \max_{j \le n} x_j$. Symmetrically, we could prove that $x_n = \min_{j \le n} x_j$, which provides the result. □

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:

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

[5] If we replace ν_β by a linear function that bounds it (e.g., $\nu_\beta \le \beta$), 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).

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

Proof. Note first that for any α and any $\boldsymbol{x}$, we have $\sum_{i=1}^{n}(A^{(\alpha)} \cdot \boldsymbol{x})_i = \sum_{i=1}^{n} x_i$ and $\sum_{i=1}^{n} g_i^{(\alpha)} = 0$. This implies that:

$$\sum_{i=1}^{n}(A^{(\alpha)} \cdot \boldsymbol{x} + \boldsymbol{g}^{(\alpha)})_i = \sum_{i=1}^{n} x_i. \tag{4}$$

Let $\overline{\boldsymbol{x}}^{(0)} = \overline{\boldsymbol{x}}$ and $\overline{\boldsymbol{x}}^{(\alpha)} = A^{(\alpha)} \cdot \overline{\boldsymbol{x}}^{(\alpha-1)} + \boldsymbol{g}^{(\alpha)}$, for $\alpha \in [1, n-\beta+1]$. We prove by induction that:

$$\sum_{i=\alpha+1}^{\alpha+\beta-1} \overline{x}_i^{(\alpha)} = \sum_{i=\alpha+1}^{\alpha+\beta-1} \overline{x}_i \quad \text{and} \quad \overline{x}_i^{(\alpha)} = \overline{x}_i \quad \text{if} \quad i \notin [\alpha+1, \alpha+\beta-1]. \quad (*)$$

This holds for $\alpha = 0$ since $\overline{\boldsymbol{x}}^{(0)} = \overline{\boldsymbol{x}}$. Let $\alpha \geq 1$. By the induction hypothesis and equality of the columns $\alpha, \ldots, \alpha+\beta-1$ of $A^{(\alpha)}$, we have $A^{(\alpha)} \cdot \overline{\boldsymbol{x}}^{(\alpha-1)} = A^{(\alpha)} \cdot \overline{\boldsymbol{x}}$ and hence $\overline{\boldsymbol{x}}^{(\alpha)} = A^{(\alpha)} \cdot \overline{\boldsymbol{x}} + \boldsymbol{g}^{(\alpha)}$. This directly implies that $\overline{x}_i^{(\alpha)} = \overline{x}_i$ when $i \notin [\alpha, \alpha+\beta-1]$. Combining this with (4) gives:

$$\sum_{i=\alpha}^{\alpha+\beta-1} \overline{x}_i^{(\alpha)} = \sum_{i=\alpha}^{\alpha+\beta-1} \overline{x}_i. \tag{5}$$

Since $\overline{x}_\alpha^{(\alpha)} = \frac{1}{2}\log \nu_\beta + \frac{1}{\beta}\sum_{j=\alpha}^{\alpha+\beta-1} \overline{x}_j^{(\alpha)}$, we obtain (using (5) and the definition of $\overline{\boldsymbol{x}}$):

$$\overline{x}_\alpha^{(\alpha)} = \frac{1}{2}\log \nu_\beta + \frac{1}{\beta}\sum_{j=\alpha}^{\alpha+\beta-1} \overline{x}_j = \overline{x}_\alpha.$$

Combining this equality and (5) allows to complete the proof of $(*)$.

It remains to prove that $\overline{x}_i^{(n-\beta+1)} = \overline{x}_i$ for $i \geq n-\beta+2$. For $i \geq n-\beta+1$, we have:

$$\overline{x}_i^{(n-\beta+1)} = \frac{1}{2}\log \nu_{n-i+1} + \frac{1}{n-i+1}\sum_{j=i}^{n} \overline{x}_j^{(n-\beta+1)}.$$

By Lemma 2 and the definition of $\boldsymbol{g}^{(n-\beta+1)}$, this implies that $\overline{x}_i^{(n-\beta+1)} = \frac{1}{\beta}\sum_{j=n-\beta+1}^{n} \overline{x}_j^{(n-\beta+1)} + g_i^{(n-\beta+1)}$. As a consequence (using (5) and the definition of $\overline{\boldsymbol{x}}$):

$$\overline{x}_i^{(n-\beta+1)} = \frac{1}{\beta}\sum_{j=n-\beta+1}^{n} \overline{x}_j + g_i^{(n-\beta+1)} = \frac{1}{\beta}\sum_{j=n-\beta+1}^{n} g_j^{(n-\beta+1)} + g_i^{(n-\beta+1)} = g_i^{(n-\beta+1)} = \overline{x}_i.$$

Overall, we have proven that $A \cdot \overline{\boldsymbol{x}} + \boldsymbol{g} = \overline{\boldsymbol{x}}^{(n-\beta+1)} = \overline{\boldsymbol{x}}$. □

Fact. Given $M_k \in \mathbb{R}^{k\times k}$, $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^k$ and $c \in \mathbb{R}$, we define $M_n \in \mathbb{R}^{n\times n}$ for $n \geq k$, as follows:

$$M_n = \left(\begin{array}{ccc|c} c & \cdots & c & \cdots\ \boldsymbol{a}^T\ \cdots \\ \vdots & \ddots & \vdots & \vdots \\ c & \cdots & c & \cdots\ \boldsymbol{a}^T\ \cdots \\ \hline \vdots & & \vdots & \\ \boldsymbol{b} & \cdots & \boldsymbol{b} & M_k \\ \vdots & & \vdots & \end{array}\right)$$

Then, for any $n \geq k$, we have $\chi(M_n)(t) = (n-k)t^{n-k-1} \cdot \chi(M_{k+1}) - (n-k-1)t^{n-k} \cdot \chi(M_k)$.

Proof of the fact. We prove the result by induction. It clearly holds for $n = k$ and $n = k + 1$. Assume now that $n > k + 1$. We have:

$$\chi(M_n)(t) = \left|\begin{array}{cc|c} (t-c) & -c \;\cdots -c & \cdots\; -\boldsymbol{a}^T \;\cdots \\ -c & (t-c)\cdots -c & \cdots\; -\boldsymbol{a}^T \;\cdots \\ \vdots & \vdots & \\ -c & -c & \\ \hline \vdots & \vdots & tI_{n-2} - M_{n-2} \\ -\boldsymbol{b} & -\boldsymbol{b} & \\ \vdots & \vdots & \end{array}\right| = \left|\begin{array}{cc|c} 2t & -t & 0 \\ -t & (t-c) & \cdots\; -c \;\cdots\; -\boldsymbol{a}^T \;\cdots \\ \hline & \vdots & \\ & -c & \\ \hline 0 & \vdots & tI_{n-2} - M_{n-2} \\ & -\boldsymbol{b} & \\ & \vdots & \end{array}\right|$$
$$= 2t \cdot \chi(M_{n-1}) - t^2 \cdot \chi(M_{n-2}).$$

The result follows by elementary calculations. □

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 $\left(\frac{n-i}{\beta-1} - \frac{3}{2}\right) \log \nu_\beta \leq \overline{x}_i - \overline{x}_{n-\beta+1} \leq \frac{n-i}{\beta-1} \log \nu_\beta$.*

Proof. We prove these bounds by induction on i for $i = n - \beta, \ldots, 1$. Recall that

$$\forall i \leq n - \beta, \quad \overline{x}_i = \frac{\beta}{2(\beta-1)} \log \nu_\beta + \frac{1}{\beta-1} \sum_{j=i+1}^{i+\beta-1} \overline{x}_j.$$

We first consider the upper bound on $\overline{x}_i - \overline{x}_{n-\beta+1}$. Since we defined Hermite's constant so that $(\nu_i)_i$ is increasing, we have $\overline{x}_{n-\beta+1} \geq \cdots \geq \overline{x}_n$. Therefore:

$$\forall i > n - \beta, \; \overline{x}_i - \overline{x}_{n-\beta+1} \leq 0 \leq \tfrac{n-i}{\beta-1} \log \nu_\beta.$$

Using the induction hypothesis, we obtain:

$$\overline{x}_i \leq \frac{\beta}{2(\beta-1)} \log \nu_\beta + \frac{1}{\beta-1} \sum_{j=i+1}^{i+\beta-1} \left(\frac{n-j}{\beta-1} \log \nu_\beta + \overline{x}_{n-\beta+1} \right) = \frac{n-i}{\beta-1} \log \nu_\beta + \overline{x}_{n-\beta+1}.$$

We now consider the lower bound on $\overline{x}_i - \overline{x}_{n-\beta+1}$. It clearly holds for $i = n - \beta + 1$. We now prove it for $i \in [n - 2(\beta-1), n - \beta]$. For that specific situation, we use the identity:

$$\forall i \in [n - 2(\beta-1), n - \beta], \quad \overline{x}_i = \frac{\beta}{2(\beta-1)} \log \nu_\beta + \frac{1}{\beta-1} \left(\sum_{j=i+1}^{n-\beta} \overline{x}_j + \sum_{j=n-\beta+1}^{i+\beta-1} \overline{x}_j \right). \tag{6}$$

As $(\overline{x}_j)_j$ decreases, we have $\frac{1}{i+2\beta-n-1} \sum_{j=n-\beta+1}^{i+\beta-1} \overline{x}_j \geq \frac{1}{\beta-1} \sum_{j=n-\beta+1}^{n} \overline{x}_j = \overline{x}_{n-\beta+1} - \frac{1}{2} \log \nu_\beta$. This implies:

$$\frac{1}{i+2\beta-n-1} \sum_{j=n-\beta+1}^{i+\beta-1} \overline{x}_j \;\geq\; \overline{x}_{n-\beta+1} + \frac{\log \nu_\beta}{i+2\beta-n-1} \sum_{j=n-\beta+1}^{i+\beta-1} \left(\frac{n-j}{\beta-1} - \frac{3}{2} \right). \tag{7}$$

Using the induction hypothesis, we also have:

$$\frac{1}{n-\beta-i} \sum_{j=i+1}^{n-\beta} \overline{x}_j \;\geq\; \overline{x}_{n-\beta+1} + \frac{\log \nu_\beta}{n-\beta-i} \sum_{j=i+1}^{n-\beta} \left(\frac{n-j}{\beta-1} - \frac{3}{2} \right). \tag{8}$$

Now, plugging (7) and (8) into (6) gives:

$$\overline{x}_i \geq \frac{\beta}{2(\beta-1)}\log\nu_\beta + \overline{x}_{n-\beta+1} + \frac{\log\nu_\beta}{\beta-1}\sum_{j=i+1}^{i+\beta-1}\left(\frac{n-j}{\beta-1}-\frac{3}{2}\right) = \left(\frac{n-i}{\beta-1}-\frac{3}{2}\right)\log\nu_\beta + \overline{x}_{n-\beta+1}.$$

When $i < n-2(\beta-1)$, the proof for the lower bound is similar to that of the upper bound. □

As the set of solutions to $\boldsymbol{x} = A\cdot\boldsymbol{x}+\boldsymbol{g}$ is $\overline{\boldsymbol{x}}+\mathrm{Span}(1,\dots,1)^T$, 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 $A_n^TA_n$. 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 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 $A_n^TA_n$.

We first explicit the characteristic polynomial χ_n of $A_n^TA_n$. 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 $n \geq \beta$:*

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

Proof. We have $A_\beta^TA_\beta = A_\beta$ and $\dim\ker(A_\beta) = \beta-1$, thus $t^{\beta-1}|\chi_\beta(t)$. Since $\mathrm{Tr}(A_\beta) = 1$ we have $\chi_\beta(t) = t^\beta(t-1)$. The computation of $A_{\beta+1}^TA_{\beta+1}$ gives:

$$A_{\beta+1}^TA_{\beta+1} = \left(\begin{array}{c|c} & \vdots \\ \frac{\beta+(\beta-1)^2}{\beta^3} & \frac{\beta-1}{\beta^2} \\ & \vdots \\ \hline \cdots\ \frac{\beta-1}{\beta^2}\ \cdots & \frac{1}{\beta}\end{array}\right).$$

If $y_1+\cdots+y_\beta = 0$ and $y_{\beta+1} = 0$, then $A_{\beta+1}^TA_{\beta+1}\cdot\boldsymbol{y} = \boldsymbol{0}$, hence $\dim\ker(A_{\beta+1}^TA_{\beta+1}) \geq \beta-1$ and $t^{\beta-1}|\chi_{\beta+1}(t)$. It can be checked that $A_{\beta+1}^TA_{\beta+1}\cdot(1,\cdots,1)^T = (1,\cdots,1)^T$. Finally, since $\mathrm{Tr}(A_{\beta+1}^TA_{\beta+1}) = 1+\frac{1}{\beta^2}$ we have $\chi_{\beta+1}(t) = t^{\beta-1}(t-1)(t-\frac{1}{\beta^2})$.

For $n \geq 1$, let C_n be the $n\times n$ bottom-right corner of $A_{n+\beta-1}^TA_{n+\beta-1}$. Note that for $n,i,j > 1$, we have $c_{nij} = c_{n-1,i-1,j-1}$, which means that we can write C_n as:

$$C_n = \left(\begin{array}{c|ccc} c_{n11} & c_{n12} & \cdots & c_{n1n} \\ \hline c_{n21} & & & \\ \vdots & & C_{n-1} & \\ c_{nn1} & & & \end{array}\right).$$

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

Moreover, we have $c_{n11} = \left(\frac{\beta-1}{\beta}\right)^2 c_{n22} + \frac{1}{\beta^2}$, $c_{ni1} = \frac{\beta-1}{\beta} c_{ni2}$ and $c_{n1i} = \frac{\beta-1}{\beta} c_{n2i}$ for all $i > 1$. Subtracting $\frac{\beta-1}{\beta}$ times the second column of $tI_n - C_n$ from the first column and subtracting $\frac{\beta-1}{\beta}$ times the second row from the first row gives:

$$\chi(C_n)(t) = \left|\begin{array}{c|ccc} \frac{2\beta^2-2\beta+1}{\beta^2}t - \frac{1}{\beta^2} & -\frac{\beta-1}{\beta}t & 0 & \cdots & 0 \\ \hline -\frac{\beta-1}{\beta}t & & & & \\ 0 & & & & \\ \vdots & & & tI_{n-1} - C_{n-1} & \\ 0 & & & & \end{array}\right|.$$

By expansion on the first column and then on the first row we obtain:

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

Since the β first columns (resp. rows) of $A_{n+\beta-1}^T A_{n+\beta-1}$ are identical, we obtain, by the previous Fact, that $\chi_{n+\beta-1}(t) = \beta t^{\beta-1} \cdot \chi(C_n)(t) - (\beta-1)t^\beta \cdot \chi(C_{n-1})(t)$. This implies that the χ_n's satisfy the same second order relation as the $\chi(C_n)$'s. □

We finally 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 $\left[1 - \frac{\pi^2\beta^2}{(n-\beta)^2}, 1 - \frac{\beta^2}{2n^2}\right]$.*

Proof. Let $\overline{\chi}_n(t)$ be the polynomial $t^n \chi_n(1/t)$. Then, by Lemma 6, we have $\overline{\chi}_\beta(t) = 1 - t$, $\overline{\chi}_{\beta+1}(t) = (1-t)\left(1 - \frac{t}{\beta^2}\right)$, and, for $n \geq \beta$:

$$\begin{aligned}\overline{\chi}_{n+2}(t) &= t^{n+2}\frac{(2\beta(\beta-1)+1)\frac{1}{t} - 1}{\beta^2} \cdot \chi_{n+1}\left(\frac{1}{t}\right) - t^{n+2}\left(\frac{\beta-1}{\beta}\right)^2 \frac{1}{t^2} \cdot \chi_n\left(\frac{1}{t}\right) \\ &= \frac{(2\beta(\beta-1)+1) - t}{\beta^2} \cdot \overline{\chi}_{n+1}(t) - \left(\frac{\beta-1}{\beta}\right)^2 \cdot \overline{\chi}_n(t).\end{aligned}$$

Let $\tau(t') = 2\beta(\beta-1)(t'-1)$ and $\psi_n(t') = \left(\frac{\beta}{\beta-1}\right)^{n-\beta} \cdot \frac{\overline{\chi}_n(1-\tau(t'))}{\tau(t')}$. We have $\psi_\beta(t') = 1$, $\psi_{\beta+1}(t') = 2t' - \frac{\beta-1}{\beta}$ and, for $n \geq \beta$:

$$\begin{aligned}\psi_{n+2}(t') &= 2t'\left(\frac{\beta}{\beta-1}\right)^{n+1-\beta} \cdot \frac{\overline{\chi}_{n+1}(1-\tau(t'))}{\tau(t')} - \left(\frac{\beta}{\beta-1}\right)^{n-\beta} \cdot \frac{\overline{\chi}_n(1-\tau(t'))}{\tau(t')} \\ &= 2t' \cdot \psi_{n+1}(t') - \psi_n(t').\end{aligned}$$

As a consequence, the ψ_n's are polynomials (in t'). Now, let $(U_n)_{n\geq 0}$ be the sequence of Chebyshev polynomials of the second kind, i.e., $U_0 = 0$, $U_1 = 1$ and $U_{n+2}(t') = 2t' \cdot U_{n+1}(t') - U_n(t')$ for $n \geq 0$. These polynomials satisfy the following property:

$$\forall n \geq 0,\ \forall x \in \mathbb{R} \setminus \{2k\pi; k \in \mathbb{Z}\},\ U_n(\cos x) = \frac{\sin(nx)}{\sin x}.$$

It can be proven by induction that $\psi_n = U_{n-\beta+1} - \frac{\beta-1}{\beta}U_{n-\beta}$ for all $n \geq \beta$. By the Fact given below, this implies that there exists $t_0' \in \left[\cos\frac{\pi}{n-\beta}, \cos\frac{\pi}{2(n-\beta+1)}\right]$ such that $\psi_n(t_0') = 0$ and $\psi_n(t') > 0$ for all $t' \in (t_0', 1)$. We have $\overline{\chi}_n(1-\tau(t_0')) = \left(\frac{\beta-1}{\beta}\right)^{n-\beta} \tau(t_0')\psi_n(t_0') = 0$, hence $t_0 = (1-\tau(t_0'))^{-1}$ is a root of $\chi_n(t)$. Since

the image of $(t'_0, 1)$ by $t' \mapsto (1-\tau(t'))^{-1}$ is $(t_0, 1)$, we obtain that t_0 is the largest root of $\chi_n(t)$ smaller than 1. We now compute bounds for t_0. We have $2(n-\beta+1) \leq 2n$ so $\cos\frac{\pi}{n-\beta} \leq t'_0 \leq \cos\frac{\pi}{2n}$. It can be checked that for $u \leq \frac{\pi}{4}$, we have $\cos u \leq 1 - \frac{8}{17}u^2$, so $1 - \frac{\pi^2}{(n-\beta)^2} \leq t'_0 \leq 1 - \frac{2\pi^2}{17n^2}$. This leads to $1 + \frac{\pi^2\beta^2}{(n-\beta)^2} \geq 1 - \tau(t'_0) \geq 1 + 2\beta(\beta-1)\frac{2\pi^2}{17n^2} \geq 1 + \frac{2\pi^2\beta^2}{17n^2}$, and thus $1 - \frac{\pi^2\beta^2}{(n-\beta)^2} \leq t_0 \leq 1 - \frac{1}{2}\frac{\beta^2}{n^2}$.

To conclude, let $\phi_n(t)$ be the polynomial $\frac{\chi_n(t)}{t-1}$. By using Lemma 6, it can be checked that $\phi_n(1) = \left(\frac{\beta-1}{\beta}\right)^{n-\beta}\frac{n}{\beta}$, which implies that $\phi_n(1) \neq 0$. This proves that 1 is never a multiple root of χ_n, which completes the proof. □

Fact. Let $n \geq 2$ and $f(x) = \frac{\sin((n+1)x)}{\sin x} - \frac{\beta-1}{\beta} \cdot \frac{\sin(nx)}{\sin x}$. The smallest positive root of f belongs to $\left[\frac{\pi}{2(n+1)}, \frac{\pi}{n}\right]$.

Proof of the fact. Since sin is an increasing function on $\left[0, \frac{\pi}{2}\right]$, we have $\sin(nx) < \sin((n+1)x)$ for all $0 < x \leq \frac{\pi}{2(n+1)}$. This implies that $f(x) > 0$ on this interval. We also have $f\left(\frac{\pi}{n}\right) = -1 < 0$. The result follows from the intermediate value theorem. □

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 $k \geq C\frac{n^2}{\beta^2}(\log n + \log\log\max_i\frac{\|\boldsymbol{b}_i^*\|}{(\det L)^{1/n}})$.

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.*

Proof. Let $\alpha \leq n-\beta+1$. We define $\widetilde{A}^{(\alpha)} = PA^{(\alpha)}P^{-1}$ and $\widetilde{\boldsymbol{g}}^{(\alpha)} = P\cdot\boldsymbol{g}^{(\alpha)}$. Let $B^{(\alpha)}$ be the basis after the first α calls to β-HKZ (starting with indices $1,\ldots,\alpha$). We first prove that we have:

$$\boldsymbol{\mu}^{(B^{(\alpha)})} \leq \widetilde{A}^{(\alpha)}\cdot\boldsymbol{\mu}^{(B^{(\alpha-1)})} + \widetilde{\boldsymbol{g}}^{(\alpha)}. \tag{9}$$

This vectorial inequality can be checked by making $\widetilde{A}^{(\alpha)}$ and $\widetilde{\boldsymbol{g}}^{(\alpha)}$ explicit:

$$\widetilde{A}_{ij}^{(\alpha)} = \begin{cases} 1 & \text{if } i=j \text{ with } i<\alpha \text{ or } i\geq\alpha+\beta-1 \\ \frac{\alpha-1}{i}\left(1-\frac{i-\alpha+1}{\beta}\right) & \text{if } i\in[\alpha,\alpha+\beta-2] \text{ and } j=\alpha-1 \\ \frac{(\alpha+\beta-1)(i-\alpha+1)}{\beta i} & \text{if } i\in[\alpha,\alpha+\beta-2] \text{ and } j=\alpha+\beta-1 \\ 0 & \text{otherwise,} \end{cases}$$

$$\widetilde{g}_i^{(\alpha)} = \begin{cases} \frac{\beta-i+\alpha-1}{i}\log\Gamma_\beta(i-\alpha+1) & \text{if } i\in[\alpha,\alpha+\beta-2] \\ 0 & \text{otherwise.} \end{cases}$$

We provide more details on the proof of (9) in appendix.

Now, let $\boldsymbol{\nu}^{(0)} = \boldsymbol{\mu}^{(B^{(0)})} = \boldsymbol{\mu}$ and $\boldsymbol{\nu}^{(\alpha)} = \widetilde{A}^{(\alpha)}\cdot\boldsymbol{\nu}^{(\alpha-1)} + \widetilde{\boldsymbol{g}}^{(\alpha)}$. We prove by induction that $\boldsymbol{\mu}^{(B^{(\alpha)})} \leq \boldsymbol{\nu}^{(\alpha)}$. For $\alpha \geq 1$, we have (successively using (9), the induction hypothesis and the fact that $\widetilde{A}^{(\alpha)} \geq 0$):

$$\boldsymbol{\mu}^{(B^{(\alpha)})} \leq \widetilde{A}^{(\alpha)}\cdot\boldsymbol{\mu}^{(B^{(\alpha-1)})} + \widetilde{\boldsymbol{g}}^{(\alpha)} \leq \widetilde{A}^{(\alpha)}\cdot\boldsymbol{\nu}^{(\alpha-1)} + \widetilde{\boldsymbol{g}}^{(\alpha)} = \boldsymbol{\nu}^{(\alpha)}.$$

The result follows, by taking $\alpha = d-\beta+1$. □

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^TA. 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:*

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

Proof. First, by using Lemma 9 and noting that $\widetilde{A} \cdot \boldsymbol{\mu}^\infty = \boldsymbol{\mu}^\infty + \widetilde{\boldsymbol{g}}$, it can be shown by induction that

$$\boldsymbol{\mu}^{(B^{[k]})} - \boldsymbol{\mu}^\infty \leq \widetilde{A}^k \cdot (\boldsymbol{\mu}^{(B^{[0]})} - \boldsymbol{\mu}^\infty). \tag{10}$$

Now, we have $\|\widetilde{A}^k \cdot (\boldsymbol{\mu}^{(B^{[0]})} - \boldsymbol{\mu}^\infty)\|_2 = \|PA^kP^{-1} \cdot (\boldsymbol{\mu}^{(B^{[0]})} - \boldsymbol{\mu}^\infty)\|_2 \leq \|P\|_2 \cdot \|A^k \cdot (\boldsymbol{x}^{(B^{[0]})} - \boldsymbol{x}^\infty)\|_2$. Thanks to the assumption on $\mu_n^{(B)}$, we know that $\boldsymbol{x}^{(B^{[0]})} - \boldsymbol{x}^\infty \in \mathrm{Span}(1, \ldots, 1)^\perp$, which is stable under A. As in theorem 2, we introduce the restriction $A_\mathcal{E}$ of A to this subspace. By the results of Subsection 4.3, we know that the largest eigenvalue of $A_\mathcal{E}^T \cdot A_\mathcal{E}$ is $\leq (1 - \frac{\beta^2}{2n^2})$. Therefore:

$$\begin{aligned}\|\widetilde{A}^k \cdot (\boldsymbol{\mu}^{(B^{[0]})} - \boldsymbol{\mu}^\infty)\|_2 &\leq \|P\|_2 \cdot \|A_\mathcal{E}^k \cdot (\boldsymbol{x}^{(B^{[0]})} - \boldsymbol{x}^\infty)\|_2 \leq \|P\|_2 \cdot \|A_\mathcal{E}\|_2^k \cdot \|\boldsymbol{x}^{(B^{[0]})} - \boldsymbol{x}^\infty\|_2 \\ &\leq \rho(P^TP)^{1/2} \cdot \left(1 - \frac{\beta^2}{2n^2}\right)^{k/2} \cdot \|\boldsymbol{x}^{(B^{[0]})} - \boldsymbol{x}^\infty\|_2,\end{aligned}$$

where ρ denotes the spectral radius. Now, the sum of the coordinates of any row of P^TP is $\leq \sum_{i=1}^n \frac{1}{i} \leq 1 + \ln n \leq 1 + \log n$. This gives $\rho(P^TP) \leq 1 + \log n$. The result follows. □

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:*

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

Proof. Wlog we assume that $\mu_n(B^{[0]}) = \mu_n^\infty$. The proof is similar to that of theorem 2. We know that:

$$\mu_1^\infty - \mu_n^\infty = x_1^\infty - \frac{1}{n}(x_1^\infty + \cdots + x_n^\infty) \leq \left(\frac{n-1}{2(\beta-1)} + \frac{3}{2}\right) \log \nu_\beta \tag{11}$$

We have $\log\left(\frac{(1+\log n)^{\frac{1}{2}} \|\boldsymbol{x}(B^{[0]}) - \boldsymbol{x}^\infty\|_2}{\log(1+\varepsilon)}\right) = O(\log \frac{n}{\varepsilon} + \log\log \max_i \|\boldsymbol{b}_i\|)$ so there exists $C \geq 0$ (independent of β) such that for any $k \geq C\frac{n^2}{\beta^2}(\log \frac{n}{\varepsilon} + \log\log \max \|\boldsymbol{b}_i\|)$, we have:

$$(1 + \log n)^{\frac{1}{2}} \left(1 - \tfrac{\beta^2}{2n^2}\right)^{\frac{k}{2}} \|x(B^{[0]}) - \boldsymbol{x}^\infty\|_2 \leq \log(1 + \varepsilon).$$

This gives $\mu_1(B^{[k]}) \leq \mu_1^\infty + \log(1 + \varepsilon) \leq \left(\mu_n(B^{[0]}) + \frac{n-1}{2(\beta-1)} + \frac{3}{2}\right) \log \nu_\beta + 1$. Taking the exponential (in base 2) leads to the result. □

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 [39,42]) 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 $\mathcal{P}oly(n) \cdot \widetilde{O}(\operatorname{size}(B))$, a basis $(\boldsymbol{b}'_i)_{i\leq n}$ such that*

$$\forall i \leq n,\ \|\boldsymbol{b}_i'^*\| \leq (1+\varepsilon)\left(\frac{4}{3}\right)^{\frac{n-i}{2}} \cdot \left(\prod_{j=i}^{n} \|\boldsymbol{b}_i'^*\|\right)^{\frac{1}{n-i+1}}. \tag{12}$$

Proof. We first prove that (12) holds for the output of Iterated-BKZ'$_2$. The remark at the end of Section 5 shows that (12) holds for $i = 1$ after the first step of the algorithm. The following steps do not modify the first vector of the basis, nor do they modify the right hand side of (12), hence the inequality holds. Now, Iterated-BKZ'$_2$ starting from Step 2 is equivalent to applying Iterated-BKZ'$_2$ to the basis $(\boldsymbol{b}_i^{(2)})_{2\leq i\leq n}$. It follows from the case $i = 1$ and a direct induction that (12) holds for all i.

We turn to analyzing the complexity. First, note that HKZ in dimension 2, i.e., Gauss' reduction, can be performed in time $\widetilde{O}(\operatorname{size}(C))$ given basis $C \in \mathbb{Q}^{2\times 2}$ as input (see [39,42]). Standard techniques allow one to bound the bit-sizes of all the vectors occurring during an execution of BKZ$'_2$ (and hence Iterated-BKZ'$_2$), by a linear function of the bit-size of the input. This completes the proof. □

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 (12), 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 $\leq 2^{O(n^2+\operatorname{size}(B))}$. We can thus find a constant c such that, for all i, the quantity $|p_i/q_i - 1|$ is either 0 or $\geq 2^{-c(n^2+\operatorname{size}(B))}$. Hence, if we choose $\varepsilon < \frac{1}{2n}.2^{-c(n^2+\operatorname{size}(B'))}$, all the inequalities from (12) must hold with $\varepsilon = 0$. Overall, we obtain, in polynomial time and using only swaps and size-reductions, a basis for which (12) 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 $\mathcal{P}oly(n)\cdot \widetilde{O}(\operatorname{size}(B))$. 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 $\mathcal{P}oly(n) \cdot \widetilde{O}(\operatorname{size}(B))$ 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 $\mathcal{P}oly(n)\cdot\widetilde{O}(\operatorname{size}(B))$.

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:

$$\forall i \leq n,\ \|\boldsymbol{c}_i^*\| \leq (1+\varepsilon)^2(4/3)^{n-i}\lambda_1(L[(\boldsymbol{b}_j^{(i)})_{i\leq j\leq n}]). \tag{13}$$

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

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 $\mathcal{P}oly(n)\cdot\widetilde{O}(\mathrm{size}(B))$ 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 $\mathcal{P}oly(n)\cdot\widetilde{O}(\mathrm{size}(C))$ bit operations. Let $(\boldsymbol{c}'_i)_{i\leq n}$ be the output basis. As size-reductions can be performed in time $\mathcal{P}oly(n)\cdot\widetilde{O}(\mathrm{size}(C)))$, 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 $\mathrm{Span}(\boldsymbol{c}_j)_{j<i}$. 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 (13), 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. M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proc. of STOC*, pages 99–108. ACM, 1996.
2. M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proc. of STOC*, pages 601–610. ACM, 2001.
3. A. Akhavi. Worst-case complexity of the optimal LLL algorithm. In *Proceedings of the 2000 Latin American Theoretical Informatics conference (LATIN 2000)*, volume 1776 of *LNCS*, pages 355–366. Springer, 2000.
4. D. Cadé, X. Pujol, and D. Stehlé. fplll-3.1, a floating-point LLL implementation. `http://perso.ens-lyon.fr/damien.stehle`.
5. D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10(4):233–260, 1997.
6. S. Galbraith. *Mathematics of Public Key Cryptography, Version 0.9.* 2011. Available at `http://www.math.auckland.ac.nz/~sgal018/crypto-book/crypto-book.html`.
7. N. Gama, N. Howgrave-Graham, H. Koy, and P. Q. Nguyen. Rankin's constant and blockwise lattice reduction. In *Proc. of CRYPTO*, number 4117 in LNCS, pages 112–130. Springer, 2006.
8. N. Gama and P. Q. Nguyen. Finding short lattice vectors within Mordell's inequality. In *Proc. of STOC*, pages 207–216. ACM, 2008.
9. N. Gama and P. Q. Nguyen. Predicting lattice reduction. In *Proceedings of Eurocrypt 2008*, volume 4965 of *LNCS*, pages 31–51. Springer, 2008.
10. O. Goldreich, S. Goldwasser, and S. Halevi. Collision-free hashing from lattice problems. Available at `http://www.eccc.uni-trier.de/`, TR96-056., 1996.
11. G. Hanrot and D. Stehlé. Improved analysis of Kannan's shortest lattice vector algorithm (extended abstract). In *Proc. of CRYPTO*, volume 4622 of *LNCS*, pages 170–186. Springer, 2007.
12. I. Haviv and O. Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In *Proc. of STOC*, pages 469–477. ACM, 2007.
13. P. S. Hirschhorn, J. Hoffstein, N. Howgrave-Graham, and W. Whyte. Choosing NTRUEncrypt parameters in light of combined lattice reduction and MITM approaches. In *Proc. of ACNS*, volume 5536 of *LNCS*, pages 437–455. Springer, 2009.

14. J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: a ring based public key cryptosystem. In *Proc. of ANTS*, volume 1423 of *LNCS*, pages 267–288. Springer, 1998.
15. R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proc. of STOC*, pages 99–108. ACM, 1983.
16. A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann*, 261:515–534, 1982.
17. H. W. Lenstra, Jr. Flags and lattice basis reduction. In *Proceedings of the third European congress of mathematics, volume 1*. Birkhäuser, 2001.
18. L. Lovász. *An Algorithmic Theory of Numbers, Graphs and Convexity*. SIAM, 1986. CBMS-NSF Regional Conference Series in Applied Mathematics.
19. M.G. Madritsch and B. Vallée. Modelling the LLL algorithm by sandpiles. In *Proc. of LATIN*, volume 6034 of *LNCS*, pages 267–281. Springer, 2010.
20. J. Martinet. *Perfect Lattices in Euclidean Spaces*. Springer, 2002.
21. D. Micciancio and O. Regev. Lattice-based cryptography. In *Post-Quantum Cryptography, D. J. Bernstein, J. Buchmann, E. Dahmen (Eds)*, pages 147–191. Springer, 2009.
22. D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. In *Proc. of STOC*, pages 351–358. ACM, 2010.
23. D. Micciancio and P. Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proc. of SODA*. ACM, 2010.
24. P. Q. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97. In *Proc. of CRYPTO*, volume 1666 of *LNCS*, pages 288–304. Springer, 1999.
25. P. Q. Nguyen and D. Stehlé. LLL on the average. In *Proc. of ANTS*, LNCS, pages 238–256. Springer, 2006.
26. P. Q. Nguyen and D. Stehlé. An LLL algorithm with quadratic complexity. *SIAM J. Comput*, 39(3):874–903, 2009.
27. P. Q. Nguyen and J. Stern. The two faces of lattices in cryptology. In *Proceedings of the 2001 Cryptography and Lattices Conference (CALC'01)*, volume 2146 of *LNCS*, pages 146–180. Springer, 2001.
28. P. Q. Nguyen and B. Vallée (editors). *The LLL Algorithm: Survey and Applications*. Information Security and Cryptography. Springer, 2009.
29. A. Novocin, D. Stehlé, and G. Villard. An LLL-reduction algorithm with quasi-linear time complexity, 2011. To appear in the proceedings of STOC. Available at `http://prunel.ccsd.cnrs.fr/ensl-00534899/en`.
30. X. Pujol and D. Stehlé. Rigorous and efficient short lattice vectors enumeration. In *Proc. of ASIACRYPT*, volume 5350 of *LNCS*, pages 390–405. Springer, 2008.
31. O. Regev. The learning with errors problem, 2010. Invited survey in CCC 2010, available at `http://www.cs.tau.ac.il/~odedr/`.
32. C. P. Schnorr. Progress on LLL and lattice reduction. Chapter of [28].
33. C. P. Schnorr. A hierarchy of polynomial lattice basis reduction algorithms. *Theor. Comput. Science*, 53:201–224, 1987.
34. C. P. Schnorr. Block reduced lattice bases and successive minima. *Combinatorics, Probability and Computing*, 3:507–533, 1994.
35. C. P. Schnorr. Accelerated slide- and LLL-reduction. *Electronic Colloquium on Computational Complexity (ECCC)*, 11(50), 2011.
36. C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. In *Proceedings of the 1991 Symposium on the Fundamentals of Computation Theory (FCT'91)*, volume 529 of *LNCS*, pages 68–85. Springer, 1991.
37. C. P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Mathematics of Programming*, 66:181–199, 1994.
38. C. P. Schnorr and H. H. Hörner. Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In *Proc. of Eurocrypt*, volume 921 of *LNCS*, pages 1–12. Springer, 1995.
39. A. Schönhage. Fast reduction and composition of binary quadratic forms. In *Proceedings of the 1991 International Symposium on Symbolic and Algebraic Computation (ISSAC'91)*, pages 128–133. ACM, 1991.
40. V. Shoup. NTL, Number Theory C++ Library. `http://www.shoup.net/ntl/`.
41. S. Wu and L. Debnath. Inequalities for convex sequences and their applications. *Computers & Mathematics with Applications*, 54(4):525–534, 2007.
42. C. K. Yap. Fast unimodular reduction: planar integer lattices. In *Proceedings of the 1992 Symposium on the Foundations of Computer Science (FOCS 1992)*, pages 437–446. IEEE Computer Society Press, 1992.

A Bounding the number of tours in the original BKZ algorithm

A bound $(n\beta)^n$ is claimed in [9]. The authors kindly explained to us how to prove a similar upper bound. We give the proof, for the sake of completeness.

First, note that during the execution of BKZ (Algorithm 1), the basis $(\boldsymbol{b}_i^{(k)})_{k\leq i\leq \min(k+\beta-1,n)}$ given as input to the SVP solver is always LLL-reduced. Now, we *modify* the call to LLL following the call to the

SVP, as follows. If the SVP solver did not find a sufficiently short vector (i.e., $\delta \cdot \|\boldsymbol{b}_k^*\| \leq \|\boldsymbol{b}\|$ in Algorithm 1), then we proceed as in Algorithm 1. Otherwise, we first call LLL on $\boldsymbol{b}, \boldsymbol{b}^{(k)}, \boldsymbol{b}_k^{(k)}, \ldots, \boldsymbol{b}_{\min(k+\beta-1,n)}^{(k)}$ to remove the linear dependency, we apply the appropriate transformation matrix to $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$, and then we call LLL again on the vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{\min(k+\beta,n)}$.

Suppose the call to the SVP solver is successful. The modification above ensures that the projected basis $\boldsymbol{b}_k^{(k)}, \ldots, \boldsymbol{b}_{\min(k+\beta-1,n)}^{(k)}$ is reduced both before the call to the SVP solver and before the second call to LLL. Furthermore, by a standard property of LLL, the vector found by the SVP solver is the first vector of the basis before the second call to LLL. Overall, the effect on the $\|\boldsymbol{b}_i^*\|$'s of a call to the SVP solver and the first call to LLL is as follows:

- $\|\boldsymbol{b}_k^*\|$ decreases by a factor $\leq \delta$,
- $\|\boldsymbol{b}_j^*\|$ remains constant if $j \notin [k, \min(k+\beta-1, n)]$,
- $\|\boldsymbol{b}_j^*\|$ does not increase by a factor $\geq 2^\beta$ if $j \in [k+1, \min(k+\beta-1, n)]$ (because the former and new $\|\boldsymbol{b}_j^*\|$'s approximate the successive minima of $L[(\boldsymbol{b}_i^{(k)})_{k \leq i \leq \min(k+\beta-1,n)}]$ (see, e.g., [6, Th. 18.12.1]).

To conclude, consider the quantity $\prod_{i \leq n} \|\boldsymbol{b}_i^*\|^{[\frac{3\beta}{\log(1/\delta)}]^{n-i+1}}$. From the above, it always decreases by a factor $\leq \frac{1}{2}$ during a successful call to the SVP solver followed by the first call to LLL. It also always decreases during a LLL swap (see [16]). Finally, it never increases during the execution of BKZ. As the input and output bases of BKZ are LLL-reduced, it always belongs to the interval

$$\left[\prod_{i \leq n} (\lambda_i 2^{-n})^{[\frac{3\beta}{\log(1/\delta)}]^{n-i+1}}, \prod_{i \leq n} (\lambda_i 2^n)^{[\frac{3\beta}{\log(1/\delta)}]^{n-i+1}}\right],$$

where the λ_i's are the successive minima of the lattice under scope. This implies that the number of calls to the SVP oracle is $O(\beta)^n$. □

B Improving the constant $\frac{3}{2}$ in Theorems 1 and 2

Theorem 1 asserts the following bound on the output of the modified BKZ algorithm:

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

We show that that there exists a universal (and efficiently computable) constant K such that for sufficiently large β and $n \geq \beta$, we have:

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

The base β of the power could be replaced by $\alpha\beta$ ($\alpha < 1$) provided that $\nu_\beta < \alpha\beta$ holds for sufficiently large β.

Proof. In the present work, we only used the facts that ν_n is an upper bound on the Hermite constant and that $\nu_n \leq \nu_{n+1}$. Since $\nu_n \leq n$, the proofs also hold with ν_n replaced by n.

Let $y_1 = 0$ and $y_{i+1} = \frac{1}{i}\sum_{j=1}^{i} y_j + \frac{i+1}{2i}\log(i+1)$ for $i \geq 2$. We have:

$$\begin{aligned}
y_{i+1} - y_i &= \frac{1}{i}\sum_{j=1}^{i-1} y_j + \frac{i+1}{2i}\log(i+1) - \frac{i-1}{i} y_i \\
&= \frac{1}{i}\sum_{j=1}^{i-1} y_j + \frac{i+1}{2i}\log(i+1) - \frac{i-1}{i}\left(\frac{1}{i-1}\sum_{j=1}^{i-1} y_j + \frac{i}{2(i-1)}\log i\right) \\
&= \frac{i+1}{2i}\log(i+1) - \frac{1}{2}\log i \\
&= \frac{1}{2}(\log(i+1) - \log i) + \frac{1}{2i}\log(i+1).
\end{aligned}$$

Let $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ be an BKZ-reduced basis and $\overline{x}_i = \log \|\boldsymbol{b}_i^*\|$. Wlog, we may assume that $\|\boldsymbol{b}_n^*\| = 1$. Under the SMA, we have $\overline{x}_i = y_{n-i+1}$ for all $i \in [n-\beta+1, n]$. We proceed as for Lemma 5: we compute upper and lower bounds for any fixed point $(\overline{x}_i)_i$ of the dynamical system $\boldsymbol{x} \leftarrow A \cdot \boldsymbol{x} + \boldsymbol{g}$. It then suffices to combine them, as in the proof of Theorem 2.

A lower bound on $\overline{x}_i$. We prove by induction on $i = n-\beta+1, \ldots, 1$ that we have:

$$\forall i \in [1, n-\beta+1], \quad \overline{x}_i - \overline{x}_{n-\beta+1} \geq \left(\frac{n-i}{\beta-1} - 1\right) \log \beta.$$

This trivially holds for $i = n-\beta+1$. As in the proof of Lemma 5, we now consider $i \in [n-2(\beta-1), n-\beta]$, for which we have (Eq. (6)):

$$\overline{x}_i = \frac{\beta}{2(\beta-1)} \log \beta + \frac{1}{\beta-1}\left(\sum_{j=i+1}^{n-\beta} \overline{x}_j + \sum_{j=n-\beta+1}^{i+\beta-1} \overline{x}_j\right). \tag{14}$$

The following sequences are concave:

- $(y_k)_{1 \leq k \leq \beta}$: It suffices to show that $y_{k+1} - y_k = \frac{1}{2}(\log(k+1) - \log k) + \frac{1}{2k}\log(k+1)$ is non-increasing; For $k \geq 3$, both $(\log(k+1) - \log k)$ and $\frac{1}{2k}\log(k+1)$ are non-increasing; It can be checked by hand that $y_4 - y_3 \leq y_3 - y_2 \leq y_2 - y_1$.
- $(y_{\beta-k+1})_{1 \leq k \leq \beta}$: By symmetry.
- $\left(\frac{1}{k}\sum_{j=1}^{k} y_{\beta-j+1}\right)_{1 \leq k \leq \beta}$: See [41, Le. 5] for example.
- $(z_k)_{k \in [1,\beta]}$ defined by $z_k = \frac{1}{k}\sum_{j=n-\beta+1}^{n-\beta+k} \overline{x}_j$: This is a simple translation of indices.

Since $(z_k)_k$ is concave, we obtain

$$z_k \geq z_1 + (z_\beta - z_1)\frac{k-1}{\beta-1} = \overline{x}_{n-\beta+1} - \frac{\log \beta}{2} \cdot \frac{k-1}{\beta-1} = \overline{x}_{n-\beta+1} - \frac{\log \beta}{k} \cdot \sum_{j=1}^{k} \frac{j-1}{\beta-1}. \tag{15}$$

Using the previous equation with $k = i + 2\beta - n - 1$ gives:

$$\frac{1}{i+2\beta-n-1} \sum_{j=n-\beta+1}^{i+\beta-1} \overline{x}_j \;\geq\; \overline{x}_{n-\beta+1} + \frac{\log \beta}{i+2\beta-n-1} \sum_{j=n-\beta+1}^{i+\beta-1} \left(\frac{n-j}{\beta-1} - 1\right). \tag{16}$$

Using the induction hypothesis (on each $\overline{x}_j$ for $j \in [n-\beta+1, i+\beta-1]$), we also have:

$$\frac{1}{n-\beta-i} \sum_{j=i+1}^{n-\beta} \overline{x}_j \;\geq\; \overline{x}_{n-\beta+1} + \frac{\log \beta}{n-\beta-i} \sum_{j=i+1}^{n-\beta} \left(\frac{n-j}{\beta-1} - 1\right). \tag{17}$$

Then, we plug (16) and (17) into (14). The end of the proof is similar to that of Lemma 5 (where the constant $\frac{3}{2}$ is replaced by 1).

An Upper bound on $\overline{x}_i$. Starting from the equation $y_{i+1} - y_i = \frac{1}{2}(\log(i+1) - \log i) + \frac{1}{2i}\log(i+1)$, we obtain:

$$\begin{aligned}
y_i &= \frac{1}{2}\log i + \sum_{j=2}^{i} \frac{\log j}{2(j-1)} \\
&= \frac{1}{2}\log i + \sum_{j=2}^{i} \left(\frac{\log j}{2j} + \frac{\log j}{2j(j-1)}\right) \\
&\leq \frac{1}{2}\log i + \int_{x=1}^{i} \frac{\log x}{2x} dx + C_1 \\
&\leq \frac{\ln 2}{4}\log^2 i + \frac{1}{2}\log i + C_2,
\end{aligned}$$

for some universal constants C_1 and C_2.

Let $f(x) = \frac{\ln 2}{4}\log^2 x + \frac{1}{2}\log x$. Let $I = [\exp(1), \beta]$. As f is concave on I, we have $f(x) \leq f(\frac{\beta}{2}) + f'(\frac{\beta}{2})(x - \frac{\beta}{2})$ for all $x \in I$ (for sufficiently large β). We have $f(\frac{\beta}{2}) \leq f(\beta) - \frac{\ln 2}{2}\log\beta + C_3$ and $|f'(\frac{\beta}{2}) - \frac{\log\beta}{\beta-1}| \leq \frac{C_4}{\beta}$ for some universal constants C_3, C_4. Since $x \mapsto \log^2 x$ is continuous and bounded on $[1, \exp(1)]$, we obtain, for any $x \in [1, \beta]$:

$$\begin{aligned} f(x) &\leq f(\beta) - \frac{\ln 2}{2}\log\beta + \frac{\log\beta}{\beta-1}(x - \frac{\beta}{2}) + C_5 \\ &= f(\beta) + \frac{\log\beta}{\beta-1}(x-\beta) + \frac{1-\ln 2}{2}\log\beta + C_6, \end{aligned}$$

for some universal constants C_5, C_6. The same holds for the y_i's (for $i \leq \beta$), as $y_i \leq f(i) + C_2$:

$$y_i \leq y_\beta + \frac{\log\beta}{\beta-1}(i-\beta) + \frac{1-\ln 2}{2}\log\beta + C_7,$$

for some universal constant C_7.

A change of variable gives the following inequality on the $\overline{x}_i$'s for $i \geq n - \beta + 1$:

$$\overline{x}_i - \overline{x}_{n-\beta+1} \leq \left(\frac{n-i}{\beta-1} - \frac{1+\ln 2}{2}\right)\log\beta + O(1).$$

It can be proved by induction that it also holds for all $i \leq n$ (as in the first part of the proof of Lemma 5). □

C Additional details for the proof of (9)

Using the following explicit value for P^{-1}

$$P^{-1} = \begin{pmatrix} 1 & & & & \\ -1 & 2 & & & \\ & -2 & 3 & & \\ & & \ddots & \ddots & \\ & & -n+1 & n \end{pmatrix},$$

it can be checked that:

$$\widetilde{A}^{(\alpha)}_{ij} = \begin{cases} 1 & \text{if } i = j \text{ with } i < \alpha \text{ or } i \geq \alpha+\beta-1 \\ \frac{\alpha-1}{i}\left(1 - \frac{i-\alpha+1}{\beta}\right) & \text{if } i \in [\alpha, \alpha+\beta-2] \text{ and } j = \alpha-1 \\ \frac{(\alpha+\beta-1)(i-\alpha+1)}{\beta i} & \text{if } i \in [\alpha, \alpha+\beta-2] \text{ and } j = \alpha+\beta-1 \\ 0 & \text{otherwise,} \end{cases}$$

$$\widetilde{\boldsymbol{g}}^{(\alpha)}_i = \begin{cases} \frac{\beta-i+\alpha-1}{i}\log \Gamma_\beta(i-\alpha+1) & \text{if } i \in [\alpha, \alpha+\beta-2] \\ 0 & \text{otherwise.} \end{cases}$$

If $i \leq \alpha-1$ or $i \geq \alpha+\beta-1$, then $\mu_i^{(B^{(\alpha)})} = (\widetilde{A}^{(\alpha)} \cdot \boldsymbol{\mu}^{(B^{(\alpha-1)})} + \widetilde{\boldsymbol{g}}^{(\alpha)})_i = \mu_i^{(B^{(\alpha-1)})}$. If $\alpha \leq i \leq \alpha+\beta-2$, we have (noting $\mu_I^{(B)} = \frac{1}{|I|}\sum_{i\in I}\log\|\boldsymbol{b}_i^*\|$ for any $I \subseteq [1,n]$):

$$\begin{aligned} i\mu_i^{(B^{(\alpha)})} &= (\alpha-1)\mu_{\alpha-1}^{(B^{(\alpha)})} + (i-\alpha+1)\mu_{[\alpha,i]}^{(B^{(\alpha)})} \\ &\leq (\alpha-1)\mu_{\alpha-1}^{(B^{(\alpha-1)})} + (\beta-i+\alpha-1)\log\Gamma_\beta(i-\alpha+1) + (i-\alpha+1)\mu_{[\alpha,\alpha+\beta-1]}^{(B^{(\alpha-1)})} \\ &= (1 - \frac{i-\alpha+1}{\beta})(\alpha-1)\mu_{\alpha-1}^{(B^{(\alpha-1)})} + (\beta-i+\alpha-1)\log\Gamma_\beta(i-\alpha+1) + \frac{(\alpha+\beta-1)(i-\alpha+1)}{\beta}\mu_{\alpha+\beta-1}^{(B^{(\alpha-1)})}. \end{aligned}$$

This completes the proof of (9). □

Efficient Public Key Encryption Based on Ideal Lattices (Extended Abstract)

Damien Stehlé[1,2], Ron Steinfeld[2], Keisuke Tanaka[3], and Keita Xagawa[3]

[1] CNRS/Department of Mathematics and Statistics, University of Sydney NSW 2006, Australia.
[2] Centre for Advanced Computing - Algorithms and Cryptography, Department of Computing, Macquarie University, NSW 2109, Australia
[3] Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Japan

Abstract. We describe public key encryption schemes with security provably based on the worst case hardness of the approximate Shortest Vector Problem in some structured lattices, called ideal lattices. Under the assumption that the latter is exponentially hard to solve even with a quantum computer, we achieve CPA-security against subexponential attacks, with (quasi-)optimal asymptotic performance: if n is the security parameter, both keys are of bit-length $\widetilde{O}(n)$ and the amortized costs of both encryption and decryption are $\widetilde{O}(1)$ per message bit. Our construction adapts the trapdoor one-way function of Gentry *et al.* (STOC'08), based on the Learning With Errors problem, to structured lattices. Our main technical tools are an adaptation of Ajtai's trapdoor key generation algorithm (ICALP'99) and a re-interpretation of Regev's quantum reduction between the Bounded Distance Decoding problem and sampling short lattice vectors.

1 Introduction

Lattice-based cryptography has been rapidly developing in the last few years, inspired by the breakthrough result of Ajtai in 1996 [1], who constructed a one-way function with average-case security provably related to the worst-case complexity of hard lattice problems. The attractiveness of lattice-based cryptography stems from its provable security guarantees, well studied theoretical underpinnings, simplicity and potential efficiency (Ajtai's one-way function is a matrix-vector multiplication over a small finite field), and also the apparent security against quantum attacks. The main complexity assumption is the hardness of approximate versions of the Shortest Vector Problem (SVP). The $\mathrm{GapSVP}_{\gamma(n)}$ problem consists in, given a lattice of dimension n and a scalar d, replying YES if there exists a non-zero lattice vector of norm $\leq d$ and NO if all non-zero lattice vectors have norm $\geq \gamma(n)d$. The complexity of $\mathrm{GapSVP}_{\gamma(n)}$ increases with n, but decreases with $\gamma(n)$. Although the latter is believed to be exponential in n for any polynomial $\gamma(n)$, minimizing the degree of $\gamma(n)$ is very important in practice, to allow the use of a practical dimension n for a given security level.

LATTICE-BASED PUBLIC KEY ENCRYPTION. The first provably secure lattice-based cryptosystem was proposed by Ajtai and Dwork [3], and relied on a variant of GapSVP in arbitrary lattices (it is now known to also rely on GapSVP [19]). Subsequent works proposed more efficient alternatives [33, 30, 9, 28]. The current state of the art [9, 28] is a scheme with public/private key length $\widetilde{O}(n^2)$ and encryption/decryption throughput of $\widetilde{O}(n)$ bit operations per message bit. Its security relies on the quantum worst-case hardness of $\mathrm{GapSVP}_{\widetilde{O}(n^{1.5})}$ in arbitrary lattices. The security can be de-quantumized at the expense of both increasing $\gamma(n)$ and decreasing the efficiency, or relying on a new and less studied problem [28]. In parallel to the provably secure schemes, there have also been heuristic proposals [11, 12]. In particular, unlike the above schemes which use unstructured random lattices, the NTRU encryption scheme [12] exploits the properties of *structured* lattices to achieve high efficiency with respect to key length ($\widetilde{O}(n)$ bits) and encryption/decryption cost ($\widetilde{O}(1)$ bit operation per message bit). Unfortunately, its security remains heuristic and it was an important open challenge to provide a provably secure scheme with comparable efficiency.

PROVABLY SECURE SCHEMES FROM IDEAL LATTICES. Micciancio [20] introduced the class of structured *cyclic* lattices, which correspond to ideals in polynomial rings $\mathbb{Z}[x]/(x^n - 1)$, and presented the first provably secure one-way function based on the worst-case hardness of the restriction of $\mathcal{P}oly(n)$-SVP to cyclic lattices. (The problem γ-SVP consists in computing a non-zero vector of a given lattice, whose norm is no more than γ times larger than the norm of a shortest non-zero lattice vector.) At the same time, thanks to its algebraic structure, this one-way function enjoys high efficiency comparable to the NTRU scheme ($\widetilde{O}(n)$ evaluation time and storage cost). Subsequently, Lyubashevsky and Micciancio [17] and independently Peikert and Rosen [29] showed how to modify Micciancio's function to construct an efficient and provably secure collision resistant hash function. For this, they introduced the more general class of *ideal* lattices, which correspond to ideals in polynomial rings $\mathbb{Z}[x]/f(x)$. The collision resistance relies on the hardness of the restriction of $\mathcal{P}oly(n)$-SVP to ideal lattices (called $\mathcal{P}oly(n)$-Ideal-SVP). The average-case collision-finding problem is a natural computational problem called Ideal-SIS, which has been shown to be as hard as the worst-case instances of Ideal-SVP. Provably secure efficient signature schemes from ideal lattices have also been proposed [18, 15, 16, 14], but constructing efficient provably secure public key encryption from ideal lattices was an interesting open problem.

OUR RESULTS. We describe the first provably CPA-secure public key encryption scheme whose security relies on the hardness of the worst-case instances of $\widetilde{O}(n^2)$-Ideal-SVP against subexponential quantum attacks. It achieves asymptotically optimal efficiency: the public/private key length is $\widetilde{O}(n)$ bits and the amortized encryption/decryption cost is $\widetilde{O}(1)$ bit operations per message bit (encrypting $\widetilde{\Omega}(n)$ bits at once, at a $\widetilde{O}(n)$ cost). Our security assumption is that $\widetilde{O}(n^2)$-Ideal-SVP cannot be solved by any subexponential time quantum algorithm, which is reasonable given the state-of-the art lattice algorithms [36]. Note that this is stronger than standard public key cryptography security as-

sumptions. On the other hand, contrary to most of public key cryptography, lattice-based cryptography allows security against subexponential quantum attacks. Our main technical tool is a re-interpretation of Regev's quantum reduction [33] between the Bounded Distance Decoding problem (BDD) and sampling short lattice vectors. Also, by adapting Ajtai's trapdoor generation algorithm [2] (or more precisely its recent improvement by Alwen and Peikert [5]) to structured ideal lattices, we are able to construct efficient provably secure trapdoor signatures, ID-based identification schemes, CCA-secure encryption and ID-based encryption. We think these techniques are very likely to find further applications.

Most of the cryptosystems based on general lattices [33, 30, 31, 9, 28] rely on the average-case hardness of the *Learning With Errors* (LWE) problem introduced in [33]. Our scheme is based on a structured variant of LWE, that we call Ideal-LWE. We introduce novel techniques to circumvent two main difficulties that arise from the restriction to ideal lattices. Firstly, the previous cryptosystems based on unstructured lattices all make use of Regev's worst-case to average-case classical reduction [33] from BDD to LWE (this is the *classical step* in the quantum reduction of [33] from SVP to LWE). This reduction exploits the unstructured-ness of the considered lattices, and does not seem to carry over to the structured lattices involved in Ideal-LWE. In particular, the probabilistic independence of the rows of the LWE matrices allows to consider a single row in [33, Cor. 3.10]. Secondly, the other ingredient used in previous cryptosystems, namely Regev's reduction [33] from the computational variant of LWE to its decisional variant, also seems to fail for Ideal-LWE: it relies on the probabilistic independence of the columns of the LWE matrices.

Our solution to the above difficulties avoids the *classical step* of the reduction from [33] altogether. Instead, we use the *quantum step* to construct a new quantum average-case reduction from SIS (the unstructured variant of Ideal-SIS) to LWE. It also works from Ideal-SIS to Ideal-LWE. Combined with the known reduction from worst-case Ideal-SVP to average-case Ideal-SIS [17], we obtain a quantum reduction from Ideal-SVP to Ideal-LWE. This shows the hardness of the computational variant of Ideal-LWE. Because we do not obtain the hardness of the decisional variant, we use a generic hardcore function to derive pseudorandom bits for encryption. This is why we need to assume the exponential hardness of SVP. The encryption scheme follows as an adaptation of [9, Sec. 7.1].

The main idea of our new quantum reduction from Ideal-SIS to Ideal-LWE is a re-interpretation of Regev's quantum step in [33]. The latter was presented as a worst-case quantum reduction from sampling short lattice vectors in a lattice L to solving BDD in the dual lattice $\hat{L}$. We observe that this reduction is actually stronger: it is an average-case reduction which works given an oracle for BDD in $\hat{L}$ with a normally distributed error vector. Also, as pointed out in [9], LWE can be seen as a BDD with a normally distributed error in a certain lattice whose dual is essentially the SIS lattice. This leads to our SIS to LWE reduction. Finally we show how to apply it to reduce Ideal-SIS to Ideal-LWE – this involves a probabilistic lower bound for the minimum of the Ideal-LWE lattice. We believe our new SIS to LWE reduction is of independent interest. Along with [22], it

provides an alternative to Regev's quantum reduction from GapSVP to LWE. Ours is weaker because the derived GapSVP factor increases with the number of LWE samples, but it has the advantage of carrying over to the ideal case. Also, when choosing practical parameters for lattice-based encryption (see, e.g., [23]), it is impractical to rely on the worst-case hardness of SVP. Instead, the practical average-case hardness of LWE is evaluated based on the best known attack which consists in solving SIS. Our reduction justifies this heuristic by showing that it is indeed necessary to (quantumly) break SIS in order to solve LWE.

ROAD-MAP. We provide some background in Section 2. Section 3 shows how to hide a trapdoor in the adaptation of SIS to ideal lattices. Section 4 contains the new reduction between SIS and LWE. Finally, in Section 5, we present our CPA-secure encryption scheme and briefly describe other cryptographic constructions.

NOTATION. Vectors will be denoted in bold. We denote by $\langle \cdot, \cdot \rangle$ and $\| \cdot \|$ the inner product and the Euclidean norm. We denote by $\rho_s(\boldsymbol{x})$ (resp. ν_s) the standard n-dimensional Gaussian function (resp. distribution) with center $\mathbf{0}$ and variance s, i.e., $\rho_s(\boldsymbol{x}) = \exp(-\pi\|\boldsymbol{x}\|^2/s^2)$ (resp. $\nu_s(\boldsymbol{x}) = \rho_s(\boldsymbol{x})/s^n$). We use the notations $\widetilde{O}(\cdot)$ and $\widetilde{\Omega}(\cdot)$ to hide poly-logarithmic factors. If D_1 and D_2 are two probability distributions over a discrete domain E, their statistical distance is $\Delta(D_1, D_2) = \frac{1}{2}\sum_{x\in E} |D_1(x) - D_2(x)|$. If a function f over a countable domain E takes non-negative real values, its sum over an arbitrary $F \subseteq E$ will be denoted by $f(F)$. If q is a prime number, we denote by $\mathbb{Z}_q$ the field of integers modulo q. We denote by Ψ_s the reduction modulo q of ν_s.

2 Reminders and Background Results on Lattices

We refer to [21] for a detailed introduction to the computational aspects of lattices. In the present section, we remind the reader very quickly some fundamental properties of lattices that we will need. We then introduce the so-called ideal lattices, and finally formally define some computational problems.

Euclidean lattices. An n-dimensional lattice L is the set of all integer linear combinations of some linearly independent vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^n$, i.e., $L = \sum \mathbb{Z}\boldsymbol{b}_i$. The $\boldsymbol{b}_i$'s are called a basis of L. The ith minimum $\lambda_i(L)$ is the smallest r such that L contains i linearly independent vectors of norms $\leq r$. We let $\lambda_1^\infty(L)$ denote the first minimum of L with respect to the infinity norm. If $B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ is a basis, we define its norm by $\|B\| = \max \|\boldsymbol{b}_i\|$ and its fundamental parallelepiped by $P(B) = \{\sum_i c_i\boldsymbol{b}_i \mid \boldsymbol{c} \in [0,1)^n\}$. Given a basis B for lattice L and a vector $\boldsymbol{c} \in \mathbb{R}^n$, we define $\boldsymbol{c} \bmod L$ as the unique vector in $P(B)$ such that $\boldsymbol{c} - (\boldsymbol{c} \bmod L) \in L$ (the basis being implicit). For any lattice L and any $s > 0$, the sum $\rho_s(L)$ is finite. We define the lattice Gaussian distribution by $D_{L,s}(\boldsymbol{b}) = \frac{\rho_s(\boldsymbol{b})}{\rho_s(L)}$, for any $\boldsymbol{b} \in L$. If L is a lattice, its dual $\widehat{L}$ is the lattice $\{\widehat{\boldsymbol{b}} \in \mathbb{R}^n \mid \forall \boldsymbol{b} \in L, \langle \widehat{\boldsymbol{b}}, \boldsymbol{b} \rangle \in \mathbb{Z}\}$. We will use the following results.

Lemma 1 ([29, Lemma 2.11] and [27, Lemma 3.5]). *For any $\boldsymbol{x}$ in an n-dimensional lattice L and $s \geq 2\sqrt{\ln(10n)/\pi}/\lambda_1^\infty(\widehat{L})$, we have $D_{L,s}(\boldsymbol{x}) \leq 2^{-n+1}$.*

Lemma 2 ([22, Lemma 2.10]). *Given an n-dimensional lattice L, we have* $\Pr_{\boldsymbol{x}\sim D_{L,s}}[\|\boldsymbol{x}\| > s\sqrt{n}] \leq 2^{-n+1}$.

Ideal lattices. Ideal lattices are a subset of lattices with the computationally interesting property of being related to polynomials via structured matrices. The n-dimensional vector-matrix product costs $\widetilde{O}(n)$ arithmetic operations instead of $O(n^2)$. Let $f \in \mathbb{Z}[x]$ a monic degree n polynomial. For any $g \in \mathbb{Q}[x]$, there is a unique pair (q, r) with $\deg(r) < n$ and $g = qf + r$. We denote r by $g \bmod f$ and identify r with the vector $\boldsymbol{r} \in \mathbb{Q}^n$ of its coefficients. We define $\mathrm{rot}_f(r) \in \mathbb{Q}^{n\times n}$ as the matrix whose rows are the $x^i r(x) \bmod f(x)$'s, for $0 \leq i < n$. We extend that notation to the matrices A over $\mathbb{Q}[x]/f$, by applying rot_f component-wise. Note that $\mathrm{rot}_f(g_1)\mathrm{rot}_f(g_2) = \mathrm{rot}_f(g_1 g_2)$ for any $g_1, g_2 \in \mathbb{Q}[x]/f$. The strengths of our cryptographic constructions depend on the choice of f. Its quality is quantified by its expansion factor (we adapt the definition of [17] to the Euclidean norm):

$$\mathrm{EF}(f,k) = \max\left\{\frac{\|g \bmod f\|}{\|g\|} \mid g \in \mathbb{Z}[x] \setminus \{0\} \text{ and } \deg(g) \leq k\,(\deg(f) - 1)\right\},$$

where we identified the polynomial $g \bmod f$ (resp. g) with the coefficients vector. Note that if $\deg(g) < n$, then $\|\mathrm{rot}_f(g)\| \leq \mathrm{EF}(f,2) \cdot \|g\|$. We will concentrate on the polynomials $x^{2^k}+1$, although most of our results are more general. We recall some basic properties of $x^{2^k}+1$ (see [7] for the last one).

Lemma 3. *Let $k \geq 0$ and $n = 2^k$. Then $f(x) = x^n + 1$ is irreducible in $\mathbb{Q}[x]$. Its expansion factor is $\leq \sqrt{2}$. Also, for any $g = \sum_{i<n} g_i x^i \in \mathbb{Q}[x]/f$, we have $\mathrm{rot}_f(g)^T = \mathrm{rot}_f(\bar{g})$ where $\bar{g} = g_0 - \sum_{1\leq i<n} g_{n-i} x^i$. Furthermore, if q is a prime such that $2n|(q-1)$, then f has n linear factors in $\mathbb{Z}_q[x]$. Finally, if $k \geq 2$ and q is a prime with $q \equiv 3 \bmod 8$, then $f = f_1 f_2 \bmod q$ where each f_i is irreducible in $\mathbb{Z}_q[x]$ and can be written $f_i = x^{n/2} + t_i x^{n/4} - 1$ with $t_i \in \mathbb{Z}_q$.*

Let I be an ideal of $\mathbb{Z}[x]/f$, i.e., a subset of $\mathbb{Z}[x]/f$ closed under addition and multiplication by any element of $\mathbb{Z}[x]/f$. It corresponds to a sublattice of $\mathbb{Z}^n$. An *f-ideal lattice* is a sublattice of $\mathbb{Z}^n$ that corresponds to an ideal $I \subseteq \mathbb{Z}[x]/f$.

Hard lattice problems. The most famous lattice problem is SVP. Given a basis of a lattice L, it aims at finding a shortest vector in $L \setminus \{\mathbf{0}\}$. It can be relaxed by asking for a non-zero vector that is no longer than $\gamma(n)$ times a solution to SVP, for a prescribed function $\gamma(\cdot)$. The best polynomial time algorithm [4, 35] solves γ-SVP only for a slightly subexponential γ. When γ is polynomial in n, then the most efficient algorithm [4] has an exponential worst-case complexity both in time and space. If we restrict the set of input lattices to ideal lattices, we obtain the problem Ideal-SVP (resp. γ-Ideal-SVP), which is implicitly parameterized by a sequence of polynomials f of growing degrees. No algorithm is known to perform non-negligibly better for Ideal-SVP than for SVP. It is believed that no subexponential quantum algorithm solves the computational variants of SVP or Ideal-SVP in the worst case. These worst-case problems can be reduced to the following average-case problems, introduced in [1] and [9].

Definition 1. *The Small Integer Solution problem with parameters $q(\cdot)$, $m(\cdot)$, $\beta(\cdot)$ ($\mathrm{SIS}_{q,m,\beta}$) is as follows: Given n and a matrix G sampled uniformly in $\mathbb{Z}_{q(n)}^{m(n)\times n}$, find $\boldsymbol{e} \in \mathbb{Z}^{m(n)} \setminus \{\mathbf{0}\}$ such that $\boldsymbol{e}^T G = \mathbf{0} \bmod q(n)$ (the modulus being taken component-wise) and $\|\boldsymbol{e}\| \leq \beta(n)$. The Ideal Small Integer Solution problem with parameters q, m, β and f ($\text{Ideal-SIS}^f_{q,m,\beta}$) is as follows: Given n and m polynomials $g_1, \ldots, g_m$ chosen uniformly and independently in $\mathbb{Z}_q[x]/f$, find $e_1, \ldots, e_m \in \mathbb{Z}[x]$ not all zero such that $\sum_{i\leq m} e_i g_i = 0$ in $\mathbb{Z}_q[x]/f$ and $\|\boldsymbol{e}\| \leq \beta$, where $\boldsymbol{e}$ is the vector obtained by concatenating the coefficients of the e_i's.*

The above problems can be interpreted as lattice problems. If $G \in \mathbb{Z}_q^{m\times n}$, then the set $G^\perp = \{\boldsymbol{b} \in \mathbb{Z}^m \mid \boldsymbol{b}^T G = \mathbf{0} \bmod q\}$ is an m-dimensional lattice and solving SIS corresponds to finding a short non-zero vector in it. Similarly, Ideal-SIS consists in finding a small non-zero element in the $\mathbb{Z}[x]/f$-module $M^\perp(\boldsymbol{g}) = \{\boldsymbol{b} \in (\mathbb{Z}[x]/f)^m \mid \langle \boldsymbol{b}, \boldsymbol{g}\rangle = 0 \bmod q\}$, where $\boldsymbol{g} = (g_1, \ldots, g_m)$. It can be seen as a lattice problem by applying the rot_f operator. Note that the m of SIS is n times larger than the m of Ideal-SIS. Lyubashevsky and Micciancio [17] reduced Ideal-SVP to Ideal-SIS. The approximation factors in [17] are given in terms of the infinity norm. For our purposes, it is more natural to use the Euclidean norm. To avoid losing a $\sqrt{n}$ factor by simply applying the norm equivalence formula, we modify the proof of [17]. We also adapt it to handle the case where the Ideal-SIS solver has a subexponentially small success probability, at the cost of an additional factor of $\widetilde{O}(\sqrt{n})$ in the SVP approximation factor.

Theorem 1. *Suppose that f is irreducible over $\mathbb{Q}$. Let $m = \mathcal{P}oly(n)$ and $q = \widetilde{\Omega}(\mathrm{EF}(f,3)\beta m^2 n)$ be integers. A polynomial-time (resp. subexponential-time) algorithm solving $\text{Ideal-SIS}^f_{q,m,\beta}$ with probability $1/\mathcal{P}oly(n)$ (resp. $2^{-o(n)}$) can be used to solve γ-Ideal-SVP in polynomial-time (resp. subexponential-time) with $\gamma = \widetilde{O}(\mathrm{EF}^2(f,2)\beta m n^{1/2})$ (resp. $\gamma = \widetilde{O}(\mathrm{EF}^2(f,2)\beta m n)$).*

The problem LWE is dual to SIS in the sense that if $G \in \mathbb{Z}_q^{m\times n}$ is the SIS-matrix, then LWE involves the dual of the lattice $G^\perp$. We have $\widehat{G^\perp} = \frac{1}{q}L(G)$ where $L(G) = \{\boldsymbol{b} \in \mathbb{Z}^m \mid \exists \boldsymbol{s} \in \mathbb{Z}_q^n, G\boldsymbol{s} = \boldsymbol{b} \bmod q\}$.

Definition 2. *The Learning With Errors problem with parameters q, m and a distribution χ on $\mathbb{R}/[0,q)$ ($\mathrm{LWE}_{q,m;\chi}$) is as follows: Given n, a matrix $G \in \mathbb{Z}_q^{m\times n}$ sampled uniformly at random and $G\boldsymbol{s} + \boldsymbol{e} \in (\mathbb{R}/[0,q))^n$, where $\boldsymbol{s} \in \mathbb{Z}_q^n$ is chosen uniformly at random and the coordinates of $\boldsymbol{e} \in (\mathbb{R}/[0,q))^m$ are independently sampled from χ, find $\boldsymbol{s}$. The Ideal Learning With Errors problem with parameters q, m, a distribution χ on $\mathbb{R}/[0,q)$ and f ($\text{Ideal-LWE}^f_{m,q;\chi}$) is the same as above, except that $G = \mathrm{rot}_f(\boldsymbol{g})$ with $\boldsymbol{g}$ chosen uniformly in $(\mathbb{Z}_q[x]/f)^m$.*

We will use the following results on the LWE and Ideal-LWE lattices.

Lemma 4. *Let n, m and q be integers with q prime, $m \geq 5n\log q$ and $n \geq 10$. Then for all but a fraction $\leq q^{-n}$ of the G's in $\mathbb{Z}_q^{m\times n}$, we have $\lambda_1^\infty(L(G)) \geq q/4$ and $\lambda_1(L(G)) \geq 0.07\sqrt{mq}$.*

Lemma 5. *Let n, m and q be integers with $q = 3 \bmod 4$ prime and $m \geq 41 \log q$ and $n = 2^k \geq 32$. Then for all but a fraction $\leq q^{-n}$ of the $\boldsymbol{g}$'s in $(\mathbb{Z}_q[x]/f)^m$, we have $\lambda_1^\infty(L(\mathrm{rot}_f(\boldsymbol{g}))) \geq q/4$ and $\lambda_1(L(\mathrm{rot}_f(\boldsymbol{g}))) \geq 0.017\sqrt{mnq}$.*

3 Hiding a Trapdoor in Ideal-SIS

In this section we show how to hide a trapdoor in the problem Ideal-SIS. Ajtai [2] showed how to simultaneously generate a (SIS) matrix $A \in \mathbb{Z}_q^{m \times n}$ and a (trapdoor) basis $S = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m) \in \mathbb{Z}^{m \times m}$ of the lattice $A^\perp = \{\boldsymbol{b} \in \mathbb{Z}^m : \boldsymbol{b}^T A = \boldsymbol{0} \bmod q\}$, with the following properties:

1. The distribution of A is close to the uniform distribution over $\mathbb{Z}_q^{m \times n}$.
2. The basis vectors $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m$ are short.

Recently, Alwen and Peikert [5] improved Ajtai's construction in the sense that the created basis has shorter vectors: $\|S\| = \widetilde{O}(n \log q)$ with $m = \Omega(n \log q)$ and overwhelming probability and $\|S\| = O(\sqrt{n \log q})$ with $m = \Omega(n \log^2 q)$. We modify both constructions to obtain a trapdoor generation algorithm for the problem Ideal-SIS, with a resulting basis whose norm is as small as the one of [5].

Before describing the construction, we notice that the construction of [5] relies on the Hermite Normal Form (HNF), but that here there is no Hermite Normal Form for the rings under scope. We circumvent this issue by showing that except in negligibly rare cases we may use a matrix which is HNF-like.

Theorem 2. *There exists a probabilistic polynomial time algorithm with the following properties. It takes as inputs n, σ, r, an odd prime q, and integers m_1, m_2. It also takes as input a degree n polynomial $f \in \mathbb{Z}[x]$ and random polynomials $\boldsymbol{a}_1 \in (\mathbb{Z}_q[x]/f)^{m_1}$. We let $f = \prod_{i \leq t} f_i$ be the factorization of f over $\mathbb{Z}_q$. We let $\kappa = \lceil 1 + \log q \rceil$, $\Delta = \left(\prod_{i \leq t} \left(1 + \left(\frac{q}{3^r}\right)^{\deg f_i}\right) - 1\right)^{1/2}$ and $m = m_1 + m_2$. The algorithm succeeds with probability $\geq 1 - p$ over $\boldsymbol{a}_1$, where $p = (1 - \prod_{i \leq t}(1 - q^{-\deg f_i}))^\sigma$. When it does, it returns $\boldsymbol{a} = \begin{pmatrix} \boldsymbol{a}_1 \\ \boldsymbol{a}_2 \end{pmatrix} \in (\mathbb{Z}_q[x]/f)^m$ and a basis S of the lattice $\mathrm{rot}_f(\boldsymbol{a})^\perp$, such that:*

1. *The distance to uniformity of $\boldsymbol{a}$ is at most $p + m_2 \Delta$.*
2. *The quality of S is as follows:*
   - *If $m_1 \geq \max\{\sigma, \kappa, r\}$ and $m_2 \geq \kappa$, then $\|S\| \leq \mathrm{EF}(f, 2) \cdot \sqrt{2} \kappa r^{1/2} n^{3/2}$. Additionally, $\|S\| \leq \mathrm{EF}(f, 2)\sqrt{3a\kappa r \cdot n}$ with probability $1 - 2^{-a + O(\log n m_1 r)}$ for a super-logarithmic function $a = a(n) = \omega(\log n)$.*
   - *If $m_1 \geq \max\{\sigma, \kappa, r\}$ and $m_2 \geq \kappa m_1$, then $\|S\| \leq \mathrm{EF}(f, 2)(4\sqrt{nr} + 3)$.*
3. *In particular, for $f = x^{2^k} + 1$ with $k \geq 2$ and a prime q with $q \equiv 3 \bmod 8$, the following holds:*
   - *We can set $\sigma = 1$ and $r = \lceil 1 + \log_3 q \rceil$. Then, the error probability is $p = q^{-\Omega(n)}$ and the parameter Δ is $2^{-\Omega(n)}$.*
   - *If $m_1, m_2 \geq \kappa$, then $\|S\| \leq \sqrt{6a\kappa r} \cdot n = O(\sqrt{an} \log q)$ with probability $1 - 2^{-a + O(\log n m_1 r)}$ for a super-logarithmic function $a = a(n) = \omega(\log n)$.*

– *If* $m_1 \geq \kappa$ *and* $m_2 \geq \kappa m_1$, *then* $\|S\| \leq \sqrt{2}(4\sqrt{nr}+3) = O(\sqrt{n \log q})$.

In the rest of this section, we only describe the analog of the second construction of Alwen and Peikert, i.e., the case $m_2 \geq \kappa m_1$, due to lack of space.

3.1 A trapdoor for Ideal-SIS

We now construct the trapdoor for Ideal-SIS. More precisely, we want to simultaneously construct a uniform $\boldsymbol{a} \in \mathcal{R}^m$ with $\mathcal{R} = \mathbb{Z}_q[x]/f$, and a small basis S of the lattice $A^\perp$ where $A = \mathrm{rot}_f(\boldsymbol{a})$. For this, it suffices to find a basis of the module $M^\perp(\boldsymbol{a}) = \{\boldsymbol{y} \in \mathcal{R}_0^m \mid \langle \boldsymbol{y}, \boldsymbol{a} \rangle \equiv 0 \bmod q\}$, with $\mathcal{R}_0 = \mathbb{Z}[x]/f$.

The principle of the design. In the following, for two matrices X and Y, $[X|Y]$ denotes the concatenation of the columns of X followed by Y and $[X;Y]$ denotes the concatenation of the rows of X and the rows of Y.

We mainly follow the Alwen-Peikert construction. Let $m_1 \geq \sigma, r$. Let us assume that we generate random polynomials $A_1 = [a_1, \ldots, a_{m_1}]^T \in \mathcal{R}^{m_1 \times 1}$. We will construct a random matrix $A_2 \in \mathcal{R}^{m_2 \times 1}$ with a structured matrix $S \in \mathcal{R}_0^{m \times m}$ such that $SA = 0$ and S is a basis of the module $M^\perp(\boldsymbol{a})$, where $A = [A_1; A_2]$. We first construct an HNF-like basis F of the module $M^\perp(\boldsymbol{a})$ with A. Next, we construct a unimodular matrix Q such that $S = QF$ is a short basis of the module. More precisely, S has the following form:

$$S = \begin{bmatrix} V & P \\ D & B \end{bmatrix} = \underbrace{\begin{bmatrix} -I_{m_1} & P \\ 0 & B \end{bmatrix}}_{Q} \cdot \underbrace{\begin{bmatrix} H & 0 \\ U & I_{m_2} \end{bmatrix}}_{F}.$$

Note that, by setting B lower triangular with diagonal coefficients equal to 1, the matrix Q is unimodular.

In this design principle, we want $FA = 0$. Hence, we should set

$$HA_1 = 0 \text{ and } A_2 = -UA_1.$$

Notice that, in order to prove that F is a basis of $A^\perp$, it suffices to show that H is a basis of $A_1^\perp$. The first equation is satisfied by setting H be an HNF-like matrix (see below). By setting $U = G + R$, with G to be defined later on and R a random matrix, we have that A_2 is almost uniformly random in $\mathcal{R}$ by Micciancio's regularity lemma (Lemma 6). More precisely, the i-th row of R is chosen from $(\{-1, 0, 1\}^n)^r \times (\{0\}^n)^{m_1 - r}$.

Lemma 6 (Adapted from [20, Th. 4.2]). *Let* $\mathbb{F}$ *be a finite field and* $f \in \mathbb{F}[x]$ *be monic and of degree* $n > 0$. *Let* R *be the ring* $\mathbb{F}[x]/f$. *Let* $D \subseteq \mathbb{F}$ *and* $r > 0$. *For* $a_1, \ldots, a_r \in R$, *we denote by* $H(a_1, \ldots, a_r)$ *the random variable* $\sum_{i \leq r} b_i a_i \in R$ *where the* b_i*'s are degree* $< n$ *polynomials with coefficients chosen independently and uniformly in* D. *If* $U_1, \ldots, U_r$ *are independent uniform random variables in* R, *then the statistical distance to uniformity*

of $(U_1, \ldots, U_r, H(U_1, \ldots, U_r))$ *is below:*

$$\frac{1}{2}\sqrt{\prod_{i \le t}\left(1 + \left(\frac{|\mathbb{F}|}{|D|^r}\right)^{\deg f_i}\right) - 1},$$

where $f = \prod_{i \le t} f_i$ *is the factorization of* f *over* $\mathbb{F}$.

We show below how to choose P and G such that $PG = H - I_{m_1}$. With this relation, the design principle form of S therefore implies that $V = -H + P(G + R) = PR - I_{m_1}$, and $D = B(G + R)$. Our constructions for P, G, B also ensure that P, B and BG have 'small' entries so that S has 'small' entries.

A construction of H without HNF. We start with how to construct H for $A_1 = [a_1, \ldots, a_{m_1}]^T \in \mathcal{R}^{m_1 \times 1}$. Since $m_1 \ge \max\{\sigma, \kappa, r\}$, we have $a_{i^*} \in \mathcal{R}^*$ for some index i^* with probability at least $1 - p$, where $\mathcal{R}^*$ denotes the set of invertible elements of $\mathcal{R}$. For now, we set $i^* = 1$ for simplicity. Using this a_{i^*}, we can construct an HNF-like matrix H: the first row is $q\boldsymbol{e}_1$ and the i-th row is $h_i\boldsymbol{e}_1 + \boldsymbol{e}_i$ for $i = 2, \ldots, m_1$, where $\boldsymbol{e}_i$ is a row vector in $\mathcal{R}_0^{m_1}$ such that the i-the element is 1 and others are 0, and $h_i = -a_i \cdot a_1^{-1} \bmod q$ such that $h_i \in [0, q)^n$. Let $\boldsymbol{h}_i$ denote the i-th row of H. By the definition of H, $H \cdot A_1 \equiv 0 \bmod q$. Thus, each row vector $\boldsymbol{h}_i$ is in $M^{\perp}(\boldsymbol{a}_1)$, where $\boldsymbol{a}_1 = A_1$. It is obvious that $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{m_1}$ are linearly independent over $\mathcal{R}_0$. Hence, we need to only show that H is indeed the basis of $M^{\perp}(\boldsymbol{a}_1)$, but this is a routine work.

Next, we consider the case where $i^* \neq 1$. In this case, we swap rows 1 and i^* of A_1 so that $a_1 \in \mathcal{R}^*$, and call it A_1'. Applying the method above, we get a basis H' of $\Lambda^{\perp}(A_1')$. By swapping columns 1 and i^* and rows 1 and i^* of H', we get a basis H of $\Lambda^{\perp}(A_1)$. In the following, we denote by i^* the index i such that $a_i \in \mathcal{R}^*$ and $h_{i,i} = q$. Note that our strategy fails if there is no index i such that $a_i \in \mathcal{R}^*$: this is not an issue, as this occurs only with small probability.

Preliminaries of the construction. Hereafter, we set $W = BG$. We often use the matrix $T_\kappa = (t_{i,j}) \in \mathcal{R}_0^{\kappa \times \kappa}$, where $t_{i,i} = 1$, $t_{i+1,i} = -2$, and all other $t_{i,j}$'s are 0. Notice that the i-th row of T_κ^{-1} is $(2^{i-1}, 2^{i-2}, \ldots, 1, 0, \ldots, 0) \in \mathcal{R}_0^{\kappa}$.

3.2 An analogue to the second Alwen-Peikert construction

The idea of the second construction in [5] is to have G contain the rows of $H - I_{m_1}$. This helps decrease the norms of the rows of P and V. To do so, we define $B = \mathrm{diag}(T_\kappa, \ldots, T_\kappa, I_{m_2 - m_1\kappa})$. Note that $B^{-1} = \mathrm{diag}(T_\kappa^{-1}, \ldots, T_\kappa^{-1}, I_{m_2 - m_1\kappa})$.

Let $\boldsymbol{h}_j'$ denote the j-th row of $H - I_{m_1}$. Let $W = [W_1; W_2; \ldots; W_{m_1}; 0]$, where $W_j = [\boldsymbol{w}_{j,\kappa}; \ldots; \boldsymbol{w}_{j,1}] \in \mathcal{R}_0^{\kappa \times m_1}$. We compute the $\boldsymbol{w}_{j,k}$'s such that $\boldsymbol{h}_j' = \sum_k 2^{k-1} \cdot \boldsymbol{w}_{j,k}$ and the components of all $\boldsymbol{w}_{j,k}$'s are polynomials with coefficients in $\{0, 1\}$. By this construction, $T_\kappa^{-1} \cdot W_j$ contains $\boldsymbol{h}_j'$ in the last row. Then, $G = B^{-1} \cdot W$ contains rows $\boldsymbol{h}_j'$ for $j = 1, \ldots, m_1$. The matrix $P = [\boldsymbol{p}_1; \ldots; \boldsymbol{p}_{m_1}]$ picks all rows $\boldsymbol{h}_1', \ldots, \boldsymbol{h}_{m_1}'$ in G by setting $\boldsymbol{p}_j = \boldsymbol{e}_{\kappa j} \in \mathcal{R}_0^{m_2}$.

The norm of S is $\max\{\|S_1\|, \|S_2\|\}$, where $S_1 = [V|P]$ and $S_2 = [D|B]$. For simplicity, we only consider the case where $f = x^n + 1$. In the general case, the bound on $\|S\|$ involves an extra $\mathrm{EF}(f, 2)$ factor.

We have that $\|BG\|^2 = \|W\|^2 \leq n$, since the entries of $\boldsymbol{h}'_j$ are all 0 except one which is either $h_{i^*,j}$ or $q-1$. Hence, we obtain that

$$\|S_2\|^2 \leq \|D\|^2 + \|B\|^2 \leq (3\sqrt{nr} + \sqrt{n})^2 + 5 \leq (4\sqrt{nr} + 3)^2.$$

It is obvious that $\|P\| \leq 1$. Additionally, we have that $\|PR\|^2 \leq nr$. Therefore:

$$\|S_1\|^2 \leq \|V\|^2 + \|P\|^2 \leq (\sqrt{nr} + 1)^2 + 1 \leq (\sqrt{nr} + 2)^2,$$

which completes the proof of Theorem 2. □

4 From LWE to SIS

We show that any efficient algorithm solving LWE with some non-negligible probability may be used by a quantum machine to efficiently solve SIS with non-negligible probability. A crucial property of the reduction is that the matrix underlying the SIS and LWE instances is preserved. This allows the reduction to remain valid while working on Ideal-SIS and Ideal-LWE.

Theorem 3. *Let q, m, n be integers, and $\alpha \in (0,1)$ with $n \geq 32$, $\mathcal{P}oly(n) \geq m \geq 5n \log q$ and $\alpha < \min\left(\frac{1}{10\sqrt{\ln(10m)}}, 0.006\right)$. Suppose that there exists an algorithm that solves $\mathrm{LWE}_{m,q;\Psi_{\alpha q}}$ in time T and with probability $\varepsilon \geq 4m \exp\left(-\frac{\pi}{4\alpha^2}\right)$. Then there exists a quantum algorithm that solves $\mathrm{SIS}_{m,q;\frac{\sqrt{m}}{2\alpha}}$ in time $\mathcal{P}oly(T, n)$ and with probability $\frac{\varepsilon^3}{64} - O(\varepsilon^5) - 2^{-\Omega(n)}$. The result still holds when replacing LWE by Ideal-LWE^f and SIS by Ideal-SIS^f, for $f = x^n + 1$ with $n = 2^k \geq 32$, $m \geq 41 \log q$ and $q \equiv 3 \bmod 8$.*

When $\alpha = O(1/\sqrt{n})$, the reduction applies even to a subexponential algorithm for LWE (with success probability $\varepsilon = 2^{-o(n)}$), transforming it into a subexponential quantum algorithm for SIS (with success probability $\varepsilon = 2^{-o(n)}$). The reduction works also for larger $\alpha = O(1/\sqrt{\log n})$, but in this case only applies to polynomial algorithms for LWE (with success probability $\varepsilon = \Omega(1/\mathcal{P}oly(n))$).

The reduction is made of two components. First, we argue that an algorithm solving LWE provides an algorithm that solves a certain bounded distance decoding problem, where the error vector is normally distributed. In a second step, we show that Regev's quantum algorithm [32, Lemma 3.14] can use such an algorithm to construct small solutions to SIS.

4.1 From LWE to BDD

An algorithm solving LWE allows us to solve, for certain lattices, a variation of the Bounded Distance Decoding problem. In that variation of BDD, the error vector is sampled according to a specified distribution.

Definition 3. *The problem* BDD_χ *with parameter distribution* $\chi(\cdot)$ *is as follows: Given an* n*-dimensional lattice* L *and a vector* $\boldsymbol{t} = \boldsymbol{b} + \boldsymbol{e}$ *where* $\boldsymbol{b} \in L$ *and* $\boldsymbol{e}$ *is distributed according to* $\chi(n)$*, the goal is to find* $\boldsymbol{b}$*. We say that a randomized algorithm* $\mathcal{A}$ *solves* BDD_χ *for a lattice* L *with success probability* $\geq \varepsilon$ *if, for* every $\boldsymbol{b} \in L$*, on input* $\boldsymbol{t} = \boldsymbol{b} + \boldsymbol{e}$*, algorithm* $\mathcal{A}$ *returns* $\boldsymbol{b}$ *with probability* $\geq \varepsilon$ *over the choice of* $\boldsymbol{e}$ *and the randomness of* $\mathcal{A}$*.*

For technical reasons, our reduction will require a randomized BDD_χ algorithm whose behaviour is independent of the solution vector $\boldsymbol{b}$, even when the error vector is fixed. This is made precise below.

Definition 4. *A randomized algorithm* $\mathcal{A}$ *solving* BDD_χ *for lattice* L *is said to be* strongly solution-independent *(SSI) if, for every fixed error vector* $\boldsymbol{e}$*, the probability (over the randomness of* $\mathcal{A}$*) that, given input* $\boldsymbol{t} = \boldsymbol{b} + \boldsymbol{e}$ *with* $\boldsymbol{b} \in L$*, algorithm* $\mathcal{A}$ *returns* $\boldsymbol{b}$ *is independent of* $\boldsymbol{b}$*.*

We show that if we have an algorithm that solves $\mathrm{LWE}_{m,q;\Psi_{\alpha q}}$, then we can construct an algorithm solving $\mathrm{BDD}_{\nu_{\alpha q}}$ for some lattices. Moreover, the constructed BDD algorithm is SSI.

Lemma 7. *Let* q, m, n *be integers and* $\alpha \in (0,1)$*, with* $m, \log q = \mathcal{P}oly(n)$*. Suppose that there exists an algorithm* $\mathcal{A}$ *that solves* $\mathrm{LWE}_{m,q;\Psi_{\alpha q}}$ *in time* T *and with probability* $\varepsilon \geq 4m \exp\left(-\frac{\pi}{4\alpha^2}\right)$*. Then there exists* $\mathcal{S} \subseteq \mathbb{Z}_q^{m \times n}$ *of proportion* $\geq \varepsilon/2$ *and an SSI algorithm* $\mathcal{A}'$ *such that if* $G \in \mathcal{S}$*, algorithm* $\mathcal{A}'$ *solves* $\mathrm{BDD}_{\nu_{\alpha q}}$ *for* $L(G)$ *in time* $T + \mathcal{P}oly(n)$ *and with probability* $\geq \varepsilon/4$*.*

Proof. If $G \in \mathbb{Z}_q^{m \times n}$ and $\boldsymbol{s} \in \mathbb{Z}_q^n$ are sampled uniformly and if the coordinates of $\boldsymbol{e}$ are sampled according to $\Psi_{\alpha q}$, then $\mathcal{A}$ finds $\boldsymbol{s}$ with probability $\geq \varepsilon$ over the choices of $G, \boldsymbol{s}$ and $\boldsymbol{e}$ and a string w of internal random bits. This implies that there exists a subset $\mathcal{S}$ of the G's of proportion $\geq \varepsilon/2$ such that for any $G \in \mathcal{S}$, algorithm $\mathcal{A}$ succeeds with probability $\geq \varepsilon/2$ over the choices of $\boldsymbol{s}$, $\boldsymbol{e}$ and w. For any $G \in \mathcal{S}$, we have $\Pr_{\boldsymbol{s},\boldsymbol{e},w}[\mathcal{A}(G\boldsymbol{s} + \boldsymbol{e}, w) = \boldsymbol{s}] \geq \varepsilon/2$.

On input $\boldsymbol{t} = \boldsymbol{b} + \boldsymbol{e}$, algorithm $\mathcal{A}'$ works as follows: it samples $\boldsymbol{s}$ uniformly in $\mathbb{Z}_q^n$; it computes $\boldsymbol{t}' = \boldsymbol{t} + A\boldsymbol{s}$, which is of the form $\boldsymbol{t}' = G\boldsymbol{s}' + q\boldsymbol{k} + \boldsymbol{e}$, where $\boldsymbol{k} \in \mathbb{Z}^m$; it calls $\mathcal{A}$ on $\boldsymbol{t}' \bmod q$ and finds $\boldsymbol{s}'$ (with probability $\geq \varepsilon/2$); it then computes $\boldsymbol{e}' = \boldsymbol{t}' - G\boldsymbol{s}' \bmod q$ and returns $\boldsymbol{t} - \boldsymbol{e}'$. Suppose that $\mathcal{A}$ succeeds, i.e., we have $\boldsymbol{s} = \boldsymbol{s}'$. Then $\boldsymbol{e}' = \boldsymbol{e} \bmod q$. Using the standard tail bound on the continuous Gaussian and the lower bound on ε we obtain that $\boldsymbol{e}$ has a component of magnitude $\geq q/2$ with probability $\leq m \exp(-\pi/(2\alpha)^2) \leq \varepsilon/4$. The algorithm thus succeeds with probability $\geq \varepsilon/2 - \varepsilon/4 = \varepsilon/4$. □

We now show that an algorithm solving $\mathrm{BDD}_{\nu_{\alpha q}}$ can be used to solve a quantized version of it. This quantization is required for the quantum part of our reduction. The intuition behind the proof is that the discretization grid is so fine (the parameter R can be chosen extremely large) that at the level of the grid the distribution ν_s looks constant.

Lemma 8. *Let $s > 0$ and L be an n-dimensional. Suppose that there exists an SSI algorithm $\mathcal{A}$ that solves BDD_{ν_s} for L in time T and with probability ε. Then there exists an R, whose bit-length is polynomial in $T, n, |\log s|$ and the bit-size of the given basis of L, and an SSI algorithm $\mathcal{A}'$ that solves $\mathrm{BDD}_{D_{L/R,s}}$ within a time polynomial in $\log R$ and with probability $\geq \varepsilon - 2^{-\Omega(n)}$.*

At this point, we have an R of bit-length polynomial in $T, n, |\log \alpha|$ and an SSI algorithm $\mathcal{B}$ with run-time polynomial in $\log R$ that solves $\mathrm{BDD}_{D_{L(G)/R,\alpha q}}$, for any G in a subset $\mathcal{S} \subseteq \mathbb{Z}_q^{m \times n}$ of proportion $\geq \varepsilon/2$, with probability $\geq \varepsilon/4 - 2^{-\Omega(n)}$ over the random choices of $\boldsymbol{e}$ and the internal randomness w. In the following we assume that on input $\boldsymbol{t} = \boldsymbol{b} + \boldsymbol{e}$, algorithm $\mathcal{B}$ outputs $\boldsymbol{e}$ when it succeeds, rather than $\boldsymbol{b}$. We implement $\mathcal{B}$ quantumly as follows: the quantum algorithm $\mathcal{B}_Q$ maps the state $|\boldsymbol{e}\rangle |\boldsymbol{b} + \boldsymbol{e}\rangle |w\rangle$ to the state $|\boldsymbol{e} - \mathcal{B}(\boldsymbol{b} + \boldsymbol{e}, w)\rangle |\boldsymbol{b} + \boldsymbol{e}\rangle |w\rangle$.

4.2 A new interpretation of Regev's quantum reduction

We first recall Regev's quantum reduction [32, Lemma 3.14]. It uses a randomized BDD oracle $\mathcal{B}^{wc}$ that finds the closest vector in a given lattice L to a given target vector, as long as the target is within a prescribed distance $d < \frac{\lambda_1(L)}{2}$ of L (as above, we assume that $\mathcal{B}^{wc}$ returns the error vector). It returns a sample from the distribution $D_{\widehat{L}, \frac{\sqrt{n}}{\sqrt{2}d}}$. We implement oracle $\mathcal{B}^{wc}$ as a quantum oracle $\mathcal{B}_Q^{wc}$ as above. We assume $\mathcal{B}_Q^{wc}$ accepts random inputs of length ℓ.

1. Set R to be a large constant and build a quantum state which is within ℓ_2 distance $2^{-\Omega(n)}$ of the normalized state corresponding to $\sum_{w \in \{0,1\}^\ell} \sum_{\boldsymbol{x} \in \frac{L}{R}, \|\boldsymbol{x}\| < d} \rho_{\frac{d}{\sqrt{n}}}(\boldsymbol{x}) |\boldsymbol{x}\rangle |\boldsymbol{x} \bmod L\rangle |w\rangle$.
2. Apply the BDD oracle $\mathcal{B}_Q^{wc}$ to the above state to remove the entanglement and obtain a state which is within ℓ_2 distance $2^{-\Omega(n)}$ of the normalized state corresponding to $\sum_{\boldsymbol{x} \in \frac{L}{R}, \|\boldsymbol{x}\| < d} \rho_{\frac{d}{\sqrt{n}}}(\boldsymbol{x}) |\boldsymbol{0}\rangle |\boldsymbol{x} \bmod L\rangle |w\rangle$.
3. Apply the quantum Fourier transform over $\mathbb{Z}_R^n$ to the second register to obtain a state that is within ℓ_2 distance $2^{-\Omega(n)}$ of the normalized state corresponding to $\sum_{\boldsymbol{x} \in \widehat{L}, \|\boldsymbol{x}\| < \frac{n}{d}} \rho_{\frac{\sqrt{n}}{d}}(\boldsymbol{x}) \left| \boldsymbol{x} \bmod (R \cdot \widehat{L}) \right\rangle$.
4. Measure the latter to obtain a vector $\widehat{\boldsymbol{b}} \bmod R \cdot \widehat{L}$. Using Babai's algorithm [6], recover $\widehat{\boldsymbol{b}}$ and output it. Its distribution is within statistical distance $2^{-\Omega(n)}$ of $D_{\widehat{L}, \frac{\sqrt{n}}{\sqrt{2}d}}$.

We now replace the perfect oracle $\mathcal{B}_Q^{wc}$ by an imperfect one.

Lemma 9. *Suppose we are given an n-dimensional lattice L, parameters $R > 2^{2n}\lambda_n(L)$ and $s < \frac{\lambda_1(L)}{2\sqrt{2n}}$, and an SSI algorithm $\mathcal{B}$ that solves $\mathrm{BDD}_{D_{\frac{L}{R}}, s}$ for L with run-time T and success probability ε. Then there exists a quantum algorithm $\mathcal{R}$ which outputs a vector $\widehat{\boldsymbol{b}} \in \widehat{L}$ whose distribution is within distance $1 - \varepsilon^2/2 + O(\varepsilon^4) + 2^{-\Omega(n)}$ of $D_{\widehat{L}, \frac{1}{2s}}$. It finishes in time polynomial in $T + \log R$.*

Proof. The quantum algorithm $\mathcal{R}$ is Regev's algorithm above with parameter $d = \sqrt{2n}s < \frac{\lambda_1(L)}{2}$, where $\mathcal{B}_Q^{wc}$ is replaced by the quantum implementation $\mathcal{B}_Q$ of $\mathcal{B}$. We just saw that if the $\mathrm{BDD}_{D_{L/R,s}}$ oracle was succeeding with probability $1-2^{-\Omega(n)}$, then the output vector $\widehat{\boldsymbol{b}}$ would follow a distribution whose statistical distance to $D_{\widehat{L},\frac{1}{2s}}$ would be $2^{-\Omega(n)}$. To work around the requirement that the oracle succeeds with overwhelming probability, we use the notion of trace distance between two quantum states, which is an adaptation of the statistical distance (see [25, Ch. 9]). The trace distance between two (pure) quantum states $|t_1\rangle$ and $|t_2\rangle$ is $\delta(|t_1\rangle, |t_2\rangle) = \sqrt{1 - |\langle t_1|t_2\rangle|^2}$. Its most important property is that for any generalized measurement (POVM), if D_1 (resp. D_2) is the resulting probability distribution when starting from $|t_1\rangle$ (resp. $|t_2\rangle$) then $\Delta(D_1, D_2) \leq \delta(|t_1\rangle, |t_2\rangle)$. Let $|t_1\rangle$ denote the state at the end of Step 2 of Regev's algorithm when we use $\mathcal{B}^{wc}$, and let $|t_2\rangle$ denote the state that we obtain at the end of Step 2 when we use $\mathcal{B}$. We upper bound $\delta(|t_1\rangle, |t_2\rangle)$ as follows.

Since $\mathcal{B}^{wc}(\boldsymbol{x} \bmod L, w) = \boldsymbol{x}$ for $\|\boldsymbol{x}\| < d$, we have that $|t_1\rangle$ is within ℓ_2 distance (and hence trace distance) $2^{-\Omega(n)}$ of the normalized state

$$|t_1'\rangle = 2^{-\ell/2} \sum_{w\in\{0,1\}^\ell} \sum_{\boldsymbol{x}\in\frac{L}{R}} \sqrt{D^d_{L/R,s}(\boldsymbol{x})}\, |\mathbf{0}\rangle\, |\boldsymbol{x} \bmod L\rangle\, |w\rangle\,,$$

where $D^d_{L/R,s}$ denotes the normalized distribution obtained by truncating $D_{L/R,s}$ to vectors of norm $< d$. On the other hand, for the imperfect oracle $\mathcal{B}$, we have that $|t_2\rangle$ is within trace distance $2^{-\Omega(n)}$ of the normalized state

$$|t_2'\rangle = 2^{-\ell/2} \sum_{w\in\{0,1\}^\ell} \sum_{\boldsymbol{x}\in\frac{L}{R}} \sqrt{D^d_{L/R,s}(\boldsymbol{x})}\, |\boldsymbol{x} - \mathcal{B}(\boldsymbol{x} \bmod L, w)\rangle\, |\boldsymbol{x} \bmod L\rangle\, |w\rangle\,.$$

Let $S_{\mathcal{B}} = \{(\boldsymbol{x}, w) \in \frac{L}{R} \times \{0,1\}^\ell \mid \|\boldsymbol{x}\| < d \text{ and } \mathcal{B}(\boldsymbol{x} \bmod L, w) = \boldsymbol{x}\}$. Notice that, if $(\boldsymbol{x}, w) \notin S_{\mathcal{B}}$, the states $|\boldsymbol{x} - \mathcal{B}(\boldsymbol{x} \bmod L, w)\rangle\, |\boldsymbol{x} \bmod L\rangle\, |w\rangle$ and $|\mathbf{0}\rangle\, |\boldsymbol{x'} \bmod L\rangle\, |w'\rangle$ are orthogonal for all $(\boldsymbol{x'}, w')$. Furthermore, if $(\boldsymbol{x}, w) \in S_{\mathcal{B}}$, the states $|\mathbf{0}\rangle\, |\boldsymbol{x} \bmod L\rangle\, |w\rangle$ and $|\mathbf{0}\rangle\, |\boldsymbol{x'} \bmod L\rangle\, |w'\rangle$ are orthogonal for all $(\boldsymbol{x'}, w') \neq (\boldsymbol{x}, w)$ with $\|\boldsymbol{x'}\| < d$, because the mapping $\boldsymbol{x} \mapsto \boldsymbol{x} \bmod L$ is 1-1 over $\boldsymbol{x}$ of norm $< d < \lambda_1(L)/2$. It follows that $|\langle t_1'|t_2'\rangle| = \sum_{(\boldsymbol{x},w)\in S_{\mathcal{B}}} 2^{-\ell} D^d_{L/R,s}(\boldsymbol{x})$. Hence, $|\langle t_1'|t_2'\rangle|$ is equal to the probability p that $\mathcal{B}(\boldsymbol{x} \bmod L, w) = \boldsymbol{x}$, over the choices of $\boldsymbol{x}$ from the distribution $D^d_{L/R,s}$ and $\boldsymbol{w}$ uniformly random in $\{0,1\}^\ell$. By Lemma 2, using the fact that $d > \sqrt{n}s$, we have $p \geq \widehat{p} - 2^{-\Omega(n)}$, where $\widehat{p}$ is the corresponding probability when $\boldsymbol{x}$ is sampled from $D_{L/R,s}$. Finally, we have $\widehat{p} = \sum_{\boldsymbol{x}} D_{L/R,s}(\boldsymbol{x}) \Pr_w[\mathcal{B}(\boldsymbol{x} \bmod L, w) = \boldsymbol{x}]$. By the strong solution-independence of $\mathcal{B}$, we have $\Pr_w[\mathcal{B}(\boldsymbol{x} \bmod L, w) = \boldsymbol{x}] = \Pr_w[\mathcal{B}(\boldsymbol{b} + \boldsymbol{x}, w) = \boldsymbol{x}]$ for any fixed $\boldsymbol{b} \in L$. Therefore, $\widehat{p}$ is the success probability of $\mathcal{B}$ in solving $\mathrm{BDD}_{D_{L/R,s}}$, so $\widehat{p} \geq \varepsilon$ by assumption. Overall, we conclude that $\delta(|t_1\rangle, |t_2\rangle) \leq \sqrt{1 - \varepsilon^2 + 2^{-\Omega(n)}}$, and hence the output of $\mathcal{R}$ is within statistical distance $1 - \varepsilon^2/2 + O(\varepsilon^4) + 2^{-\Omega(n)}$ of $D_{\widehat{L},\frac{1}{2s}}$, as claimed. □

To prove Theorem 3, we apply Lemma 9 to the lattices $L(G)$ for $G \in \mathcal{S}$, with algorithm $\mathcal{B}$. For that, we need to ensure that the hypothesis $\alpha q < \frac{\lambda_1(L(G))}{2\sqrt{2m}}$ is

satisfied. From Lemma 4 (resp. Lemma 5 in the case of Ideal-LWE), we know that with probability $1-2^{-\Omega(n)}$ over the choice of G in $\mathbb{Z}_q^{m\times n}$, we have $\lambda_1^\infty(L(G)) \geq \frac{q}{4}$ and $\lambda_1(L(G)) \geq 0.07\sqrt{m}q$. For such 'good' G's, the hypothesis $\alpha q < \frac{\lambda_1(L(G))}{2\sqrt{2m}}$ is satisfied, since $\alpha < 0.006$. The set $\mathcal{S}'$ of the G's in $\mathcal{S}$ for which that condition is satisfied represents a proportion $\geq \varepsilon/2 - 2^{-\Omega(n)}$ of $\mathbb{Z}_q^{m\times n}$. Suppose now that $G \in \mathcal{S}'$. Lemma 9 shows that we can find a vector $\boldsymbol{s} \in G^\perp = q\widehat{L(G)}$ that follows a distribution whose distance to $D_{G^\perp,\frac{1}{2\alpha}}$ is $\Delta = 1 - \frac{\varepsilon^2}{32} + O(\varepsilon^4) + 2^{-\Omega(n)}$. Thanks to Lemmas 1 and 2 (since $G \in \mathcal{S}$ and $\alpha \leq 1/(10\sqrt{\ln(10m)})$, the hypothesis of Lemma 1 is satisfied), we have that with probability $\geq 1 - 2^{-\Omega(n)} - \Delta = \frac{\varepsilon^2}{32} - O(\varepsilon^4) - 2^{-\Omega(n)}$, the returned $\boldsymbol{s}$ is a non-zero vector of $G^\perp$ whose norm is $\leq \frac{\sqrt{m}}{2\alpha}$. Multiplying by the probability $\geq \varepsilon/2 - 2^{-\Omega(n)}$ that $G \in \mathcal{S}'$ gives the claimed success probability and completes the proof of Theorem 4. □

5 Cryptographic Applications

We now use the results of Sections 3 and 4 to construct efficient cryptographic primitives based on ideal lattices. This includes the first provably secure lattice-based public-key encryption scheme with asymptotically optimal encryption and decryption computation costs of $\widetilde{O}(1)$ bit operations per message bit.

5.1 Efficient public-key encryption scheme

Our scheme is constructed in two steps. Firstly, we use the LWE mapping $(\boldsymbol{s}, \boldsymbol{e}) \mapsto G \cdot \boldsymbol{s} + \boldsymbol{e} \bmod q$ as an injective trapdoor one-way function, with the trapdoor being the full-dimensional set of vectors in $G^\perp$ from Section 3, and the one-wayness being as hard as Ideal-SIS (and hence Ideal-SVP) by Theorem 3. This is an efficient ideal lattice analogue of some trapdoor functions presented in [9, 28] for arbitrary lattices. Secondly, we apply the Goldreich-Levin hardcore function based on Toeplitz matrices [10, Sec. 2.5] to our trapdoor function, and XOR the message with the hardcore bits to obtain a semantically secure encryption. To obtain the $\widetilde{O}(1)$ amortized bit complexity per message bit, we use $\widetilde{\Omega}(n)$ hardcore bits, which induces a subexponential loss in the security reduction.

Our trapdoor function family Id-Trap is defined in Figure 1. For security parameter $n = 2^k$, we fix $f(x) = x^n + 1$ and $q = \mathcal{P}oly(n)$ a prime satisfying $q \equiv 3 \bmod 8$. From Lemma 3, it follows that f splits modulo q into two irreducible factors of degree $n/2$. We set $\sigma = 1$, $r = 1 + \log_3 q = \widetilde{O}(1)$ and $m = (\lceil \log q \rceil + 1)\sigma + r = \widetilde{O}(1)$. We define $\mathcal{R} = \mathbb{Z}_q[x]/f$. The following lemma ensures the correctness of the scheme (this is essentially identical to [28, Sec. 4.1]) and asserts that the evaluation and inversion functions can be implemented efficiently.

Lemma 10. *Let $q > 2\sqrt{mn}L$ and $\alpha = o(1/(L\sqrt{\log n}))$. Then for any $s \in \mathcal{R}$ and for $\boldsymbol{e}$ sampled from $\overline{\Psi}_{\alpha q}$, the inversion algorithm recovers $(\boldsymbol{s}, \boldsymbol{e})$ with probability $1 - n^{-\omega(1)}$ over the choice of $\boldsymbol{e}$. Furthermore, the evaluation and inversion algorithms for $h_{\boldsymbol{g}}$ can be implemented with run-time $\widetilde{O}(n)$.*

– **Generating a function with trapdoor.** Run the algorithm from Theorem 2, using $f = x^n + 1, n, q, r, \sigma, m$ as inputs. Suppose it succeeds. It returns $\boldsymbol{g} \in (\mathbb{Z}_q[x]/f)^m$ (function index) and a trapdoor full-rank set S of linearly independent vectors in $\mathrm{rot}_f(\boldsymbol{g})^\perp \subseteq \mathbb{Z}_q^{mn \times mn}$ with $\|S\| \leq \sqrt{2}(4\sqrt{nr} + 3) =: L$ (we have $L = \widetilde{O}(\sqrt{n})$).
– **Function evaluation.** Given function index $\boldsymbol{g}$, we define the trapdoor function $h_{\boldsymbol{g}} : \mathbb{Z}_q^n \times \mathbb{Z}_q^{mn} \to \mathbb{Z}_q^{mn}$ as follows. On input $\boldsymbol{s}$ uniformly random in $\mathbb{Z}_q^n$ and $\boldsymbol{e} \in \mathbb{Z}_q^{mn}$ sampled from $\overline{\Psi}_{\alpha q}$ (defined as the rounding of $\Psi_{\alpha q}$ to the closest integer vector), we compute and return: $\boldsymbol{c} = h_{\boldsymbol{g}}(\boldsymbol{s}, \boldsymbol{e}) := \mathrm{rot}_f(\boldsymbol{g}) \cdot \boldsymbol{s} + \boldsymbol{e} \bmod q$.
– **Function inversion.** Given $\boldsymbol{c} = h_{\boldsymbol{g}}(\boldsymbol{s}, \boldsymbol{e})$ and trapdoor S, compute $\boldsymbol{d} = S^T \cdot \boldsymbol{c} \bmod q$ and $\boldsymbol{e}' = S^{-T} \cdot \boldsymbol{d}$ (in $\mathbb{Q}$). Compute $\boldsymbol{u} = \boldsymbol{c} - \boldsymbol{e}' \bmod q$ and $\boldsymbol{s}' = (\mathrm{rot}_f(g_1))^{-1} \cdot \boldsymbol{u}_1 \bmod q$, where $\boldsymbol{u}_1$ consists of the first n coordinates of $\boldsymbol{u}$. Return $(\boldsymbol{s}', \boldsymbol{e}')$.

Fig. 1. The trapdoor function family Id-Trap.

The one-wayness of Id-Trap is equivalent to the hardness of $\mathrm{LWE}_{m,q;\overline{\Psi}_{\alpha q}}$. Furthermore, an instance of $\mathrm{LWE}_{m,q;\Psi_{\alpha q}}$ can be efficiently converted by rounding to an instance of $\mathrm{LWE}_{m,q;\overline{\Psi}_{\alpha q}}$. This proves Lemma 11.

Lemma 11. *Any attacker against the one-wayness of* Id-Trap *(with parameters m, α, q) with run-time T and success probability ε provides an algorithm for* $\mathrm{LWE}_{m,q;\Psi_{\alpha q}}$ *with run-time T and success probability ε.*

By combining our trapdoor function with the GL hardcore function [10, Sec. 2.5] we get the encryption scheme of Figure 2.

– **Key generation.** For security parameter n, run the generation algorithm of Id-Trap to get an $h_{\boldsymbol{g}}$ and a trapdoor S. We can view the first component of the domain of $h_{\boldsymbol{g}}$ as a subset of $\mathbb{Z}_2^{\ell_I}$ for $\ell_I = O(n \log q) = \widetilde{O}(n)$. Generate $\boldsymbol{r} \in \mathbb{Z}_2^{\ell_I + \ell_M}$ uniformly and define the Toeplitz matrix $M_{GL} \in \mathbb{Z}_2^{\ell_M \times \ell_i}$ (allowing fast multiplication [26]) whose ith row is $[r_i, \ldots, r_{\ell_I + i - 1}]$. The public key is $(\boldsymbol{g}, \boldsymbol{r})$ and the secret key is S.
– **Encryption.** Given ℓ_M-bit message M with $\ell_M = n/\log n = \widetilde{\Omega}(n)$ and public key $(\boldsymbol{g}, \boldsymbol{r})$, sample $(\boldsymbol{s}, \boldsymbol{e})$ with $\boldsymbol{s} \in \mathbb{Z}_q^n$ uniform and $\boldsymbol{e}$ sampled from $\overline{\Psi}_{\alpha q}$, and evaluate $C_1 = h_{\boldsymbol{g}}(\boldsymbol{s}, \boldsymbol{e})$. Compute $C_2 = M \oplus (M_{GL} \cdot \boldsymbol{s})$, where the product $M_{GL} \cdot \boldsymbol{s}$ is computed over $\mathbb{Z}_2$, and $\boldsymbol{s}$ is viewed as a string over $\mathbb{Z}_2^{\ell_I}$. Return the ciphertext (C_1, C_2).
– **Decryption.** Given ciphertext (C_1, C_2) and secret key $(S, \boldsymbol{r})$, invert C_1 to compute $(\boldsymbol{s}, \boldsymbol{e})$ such that $h_{\boldsymbol{g}}(\boldsymbol{s}, \boldsymbol{e}) = C_1$, and return $M = C_2 \oplus (M_{GL} \cdot \boldsymbol{s})$.

Fig. 2. The semantically secure encryption scheme Id-Enc.

Theorem 4. *Any IND-CPA attacker against* Id-Enc *with run-time T and success probability $1/2 + \varepsilon$ provides an algorithm for* $\text{Ideal-LWE}^f_{m,q;\Psi_{\alpha q}}$ *with run-time $O(2^{3\ell_M} n^3 \varepsilon^{-3} \cdot T)$ and success probability $\Omega(2^{-\ell_M} n^{-1} \cdot \varepsilon)$.*

Proof. The attacker can be converted to a GL hardcore function distinguisher that, given $C_1 = h_{\boldsymbol{g}}(\boldsymbol{s}, \boldsymbol{e})$, M_{GL}, and ℓ_M bit string z, for $\boldsymbol{s}$ sampled uniformly in $\mathbb{Z}_q^n$, $\boldsymbol{e}$ sampled from $\overline{\Psi}_{\alpha q}$, and M_{GL} constructed as in the key generation procedure, distinguishes whether z is uniformly random (independent of $\boldsymbol{s}$ and $\boldsymbol{e}$) or $z = M_{GL} \cdot \boldsymbol{s}$. It has run-time T and advantage ε. The result follows by applying Lemma 2.5.8, Proposition 2.5.7 and Proposition 2.5.3 in [10]. Note that we do not need to give the vector $\boldsymbol{e}$ additionally to $\boldsymbol{s}$ as input to the GL function, as $\boldsymbol{e}$ is uniquely determined once $\boldsymbol{s}$ is given (with overwhelming probability). □

By using Lemma 10 and Theorems 1, 3 and 4, we get our main result.

Corollary 1. *Any IND-CPA attacker against encryption scheme* Id-Enc *with run-time* $2^{o(n)}$ *and success probability* $1/2+2^{-o(n)}$ *provides a quantum algorithm for* $\widetilde{O}(n^2)$-Ideal-SVP *with* $f(x) = x^n + 1$ *and* $n = 2^k$*, with run-time* $2^{o(n)}$ *and overwhelming success probability. Furthermore, the scheme* Id-Enc *encrypts and decrypts* $\widetilde{\Omega}(n)$ *bits within* $\widetilde{O}(n)$ *bit operations, and its keys have* $\widetilde{O}(n)$ *bits.*

5.2 Further applications

Our results have several other applications, adapting various known constructions for unstructured lattices to ideal lattices, as summarised below.

CCA2-secure encryption. Peikert [28] derived a CCA2-secure encryption scheme from the non-structured variant of the trapdoor function family Id-Trap from Figure 1, using the framework of [31, 34] for building a CCA2-secure scheme from a collection of injective trapdoor functions that is secure under correlated product (i.e., one-wayness is preserved if several functions are evaluated on the same input). The approach of [28] can be applied to Id-Trap, using the equality between Ideal-LWE$_{km}$ and the product of k instances of Ideal-LWE$_m$, multiple hardcore bits as in Id-Enc, and instantiating the required strongly unforgeable signature with the Ideal-SVP-based scheme of [18]. By choosing $k = \widetilde{O}(n)$ (the bit-length of the verification key in [18]) and $\alpha = \widetilde{O}(n^{-3/2})$, we obtain a CCA2-secure scheme that encrypts $\widetilde{\Omega}(n)$ bits within $\widetilde{O}(n^2)$ bit operations and whose security relies on the exponential quantum hardness of $\widetilde{O}(n^4)$-Ideal-SVP.

Trapdoor signatures. Gentry *et al.* [9] give a construction of a trapdoor signature (in the random oracle model) from any family of collision-resistant preimage sampleable functions (PSFs). They show how to sample preimages of $f_G(\boldsymbol{x}) = \boldsymbol{x}^T G$, where $G \in \mathbb{Z}_q^{m \times n}$, using a full-dimensional set of short vectors in $G^\perp$. By applying this to $G = \mathrm{rot}_f(\boldsymbol{g})$ and using the trapdoor generation algorithm from Section 3, we obtain a PSF whose collision resistance relies on Ideal-SIS, and hence Ideal-SVP, and thus a structured variant of the trapdoor signature scheme of [9], with $\widetilde{O}(n)$ verification time and signature length.

ID-based identification. From lattice-based signatures, we derive ID-based identification (IBI) and ID-based signature (IBS). Applying the standard strategy, we construct lattice-based IBI schemes as follows: The master generates a key pair of a lattice-based signature scheme, say (G, S); Each user obtains from the master a short vector $\boldsymbol{e}$ such that $\boldsymbol{e}^T G = H(id)$, where H is a random oracle; The prover proves to the verifier that he/she has a short vector $\boldsymbol{e}$ through the Micciancio-Vadhan protocol [24]. This combination yields concurrently secure IBI schemes based on $\widetilde{O}(n^2)$-SVP and $\widetilde{O}(n^2)$-Ideal-SVP in the random oracle model. As the MV protocol is witness indistinguishable, we can use the Fiat-Shamir heuristic [8] and derive lattice-based IBS schemes.

ID-based encryption (IBE). It is shown in [9] that the unstructured variant of the above trapdoor signature can be used as the identity key extraction for an IBE scheme. This requires a 'dual' version of Id-Enc, in which the public key

is of the form $(\boldsymbol{g}, u)$, where $u = H(id)$ is the hashed identity, and the secret key is the signature of id, i.e., a short preimage of u under $f_{\boldsymbol{g}}(\boldsymbol{x}) = \boldsymbol{x}^T \mathrm{rot}_f(\boldsymbol{g})$. We construct the 'dual' encryption as (C_1, C_2) where $C_1 = h_{\boldsymbol{g}}(\boldsymbol{s}, \boldsymbol{e})$ and $C_2 = T_\ell(\mathrm{rot}_f(u) \cdot \boldsymbol{s}) + M$, where $M \in \mathbb{Z}_q^\ell$ contains the message and $T_\ell(\mathrm{rot}_f(u) \cdot \boldsymbol{s})$ denotes the first ℓ coordinates of $\mathrm{rot}_f(u) \cdot \boldsymbol{s} \bmod q$. By adapting the results of [13], we show that $T_\ell(\mathrm{rot}_f(u) \cdot \boldsymbol{s})$ is an exponentially-secure generic hardcore function for uniform $u \in \mathbb{Z}_q^n$, when $\ell = o(n)$. This allows us to prove the IND-CPA security of the resulting IBE scheme based on the hardness of Ideal-SVP.

Acknowledgements. We thank Chris Peikert and Oded Regev for helpful discussions. The first author was partly supported by the LaRedA ANR grant, the second author by a Macquarie University Research Fellowship (MQRF) and ARC Discovery Grant DP0987734, and the fourth author by KAKENHI 19-55201.

References

1. M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of STOC 1996*, pages 99–108. ACM, 1996.
2. M. Ajtai. Generating hard instances of the short basis problem. In *Proceedings of ICALP 1999*, volume 1644 of *LNCS*, pages 1–9. Springer, 1999.
3. M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of STOC 1997*, pages 284–293. ACM, 1997.
4. M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings of STOC 2001*, pages 601–610. ACM, 2001.
5. J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. In *Proceedings of STACS 2009*, LNCS. Springer, 2009.
6. L. Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6:1–13, 1986.
7. I. F. Blake, S. Gao, and R. C. Mullin. Explicit factorization of $x^{2^k} + 1$ over F_p with prime $p \equiv 3 \bmod 4$. *App. Alg. in Eng., Comm. and Comp*, 4:89–94, 1992.
8. A. Fiat and A. Shamir. How to prove yourself – practical solutions to identification and signature problems. In *Proceedings of Crypto 1986*, volume 263 of *LNCS*, pages 186–194. Springer, 1986.
9. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of STOC 2008*, pages 197–206. ACM, 2008.
10. O. Goldreich. *Foundations of Cryptography*, volume II – Basic Applications. Cambridge University Press, 2001.
11. O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *Proceedings of Crypto 1997*, volume 1294 of *LNCS*, pages 112–131. Springer, 1997.
12. J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: a ring based public key cryptosystem. In *Proceedings of ANTS III*, volume 1423 of *LNCS*, pages 267–288. Springer, 1998.
13. T. Holenstein, U. Maurer, and J. Sjödin. Complete classification of bilinear hardcore functions. In *Proceedings of Crypto 2004*, volume 3152 of *LNCS*, pages 73–91. Springer, 2004.
14. A. Kawachi, K. Tanaka, and K. Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *Proceedings of Asiacrypt 2008*, volume 5350 of *LNCS*, pages 372–389. Springer, 2008.

15. V. Lyubashevsky. Lattice-based identification schemes secure under active attacks. In *Proceedings of PKC 2008*, volume 4939 of *LNCS*, pages 162–179. Springer, 2008.
16. V. Lyubashevsky. *Towards Practical Lattice-Based Cryptography.* PhD thesis, University of California, San Diego, 2008.
17. V. Lyubashevsky and D. Micciancio. Generalized compact knapsacks are collision resistant. In *Proceedings of ICALP 2006)*, volume 4052 of *LNCS*, pages 144–155. Springer, 2006.
18. V. Lyubashevsky and D. Micciancio. Asymptotically efficient lattice-based digital signatures. In *Proceedings of TCC 2008*, volume 4948 of *LNCS*, pages 37–54. Springer, 2008.
19. V. Lyubashevsky and D. Micciancio. On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In *Proceedings of Crypto 2009*, volume 5677 of *LNCS*, pages 450–461. Springer, 2009.
20. D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, 16(4):365–411, 2007.
21. D. Micciancio and S. Goldwasser. *Complexity of lattice problems: a cryptographic perspective.* Kluwer Academic Press, 2002.
22. D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.
23. D. Micciancio and O. Regev. *Post-Quantum Cryptography*, chapter Lattice-based Cryptography. Springer, 2008.
24. D. Micciancio and S. Vadhan. Statistical zero-knowledge proofs with efficient provers: Lattice problems and more. In *Proceedings of Crypto 2003*, volume 2729 of *LNCS*, pages 282–298. Springer, 2003.
25. M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.
26. V. Y. Pan. *Structured matrices and polynomials, unified superfast algorithms.* Springer and Birkhäuser, 2001.
27. C. Peikert. Limits on the hardness of lattice problems in ℓ_p norms. *Computational Complexity*, 2(17):300–351, 2008.
28. C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proceedings of STOC 2009*, pages 333–342. ACM, 2009.
29. C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *Proceedings of TCC 2006*, pages 145–166, 2006.
30. C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In *Proceedings of Crypto 2008*, volume 5157 of *LNCS*, pages 554–571. Springer, 2008.
31. C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *Proceedings of STOC 2008*, pages 187–196. ACM, 2008.
32. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. Extended version of [33] dated May 2, 2009. Available at the URL `http://www.cs.tau.ac.il/~odedr/`.
33. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of STOC 2005*, pages 84–93. ACM, 2005.
34. A. Rosen and G. Segev. Chosen-ciphertext security via correlated products. In *Proceedings of TCC 2009*, volume 5444 of *LNCS*, pages 419–436. Springer, 2009.
35. C. P. Schnorr. A hierarchy of polynomial lattice basis reduction algorithms. *Theor. Comput. Sci*, 53:201–224, 1987.
36. C. P. Schnorr. Hot topics of LLL and lattice reduction. To appear in the proceedings of the LLL+25 conference, 2009.

Making NTRU as Secure as Worst-Case Problems over Ideal Lattices

Damien Stehlé[1] and Ron Steinfeld[2]

[1] CNRS, Laboratoire LIP (U. Lyon, CNRS, ENS Lyon, INRIA, UCBL), 46 Allée d'Italie, 69364 Lyon Cedex 07, France.
damien.stehle@gmail.com – http://perso.ens-lyon.fr/damien.stehle
[2] Centre for Advanced Computing - Algorithms and Cryptography, Department of Computing, Macquarie University, NSW 2109, Australia
ron.steinfeld@mq.edu.au – http://web.science.mq.edu.au/~rons

Abstract. NTRUEncrypt, proposed in 1996 by Hoffstein, Pipher and Silverman, is the fastest known lattice-based encryption scheme. Its moderate key-sizes, excellent asymptotic performance and conjectured resistance to quantum computers could make it a desirable alternative to factorisation and discrete-log based encryption schemes. However, since its introduction, doubts have regularly arisen on its security. In the present work, we show how to modify NTRUEncrypt to make it provably secure in the standard model, under the assumed quantum hardness of standard worst-case lattice problems, restricted to a family of lattices related to some cyclotomic fields. Our main contribution is to show that if the secret key polynomials are selected by rejection from discrete Gaussians, then the public key, which is their ratio, is statistically indistinguishable from uniform over its domain. The security then follows from the already proven hardness of the R-LWE problem.

Keywords. Lattice-based cryptography, NTRU, provable security.

1 Introduction

NTRUEncrypt, devised by Hoffstein, Pipher and Silverman, was first presented at the Crypto'96 rump session [14]. Although its description relies on arithmetic over the polynomial ring $\mathbb{Z}_q[x]/(x^n - 1)$ for n prime and q a small integer, it was quickly observed that breaking it could be expressed as a problem over Euclidean lattices [6]. At the ANTS'98 conference, the NTRU authors gave an improved presentation including a thorough assessment of its practical security against lattice attacks [15]. We refer to [13] for an up-to-date account on the past 15 years of security and performance analyses. Nowadays, NTRUEncrypt is generally considered as a reasonable alternative to the encryption schemes based on integer factorisation and discrete logarithm over finite fields and elliptic curves, as testified by its inclusion in the IEEE P1363 standard [18]. It is also often considered as the most viable post-quantum public-key encryption (see, e.g., [31]).

In parallel to a rising number of attacks and practical improvements on NTRUEncrypt the (mainly) theoretical field of provably secure lattice-based cryptography has steadily been developed. It originated in 1996 with Ajtai's acclaimed worst-case to average-case reduction [2], leading to a collision-resistant hash function that is as hard to break as solving several worst-case problems defined over lattices. Ajtai's average-case problem is now referred to as the *Small Integer Solution* problem (SIS). Another major breakthrough in this field was the introduction in 2005 of the *Learning with Errors* problem (LWE) by Regev [32]: LWE is both hard on the average (worst-case lattice problems quantumly reduce to it), and sufficiently flexible to allow for the design of cryptographic functions. In the last few years, many cryptographic schemes have been introduced that are provably as secure as LWE and SIS are hard (and thus provably secure, assuming the worst-case hardness of lattice problems). These include CPA and CCA secure encryption schemes, identity-based encryption schemes, digital signatures, *etc* (see [32, 29, 11, 5, 1] among others, and the surveys [25, 33]).

The main drawback of cryptography based on LWE and SIS is its limited efficiency. A key typically contains a random matrix defined over $\mathbb{Z}_q$ for a small q, whose dimension is linear in the security parameter; consequently, the space and time requirements seem bound to be at least quadratic with respect to the security parameter. In 2002, Micciancio [23] succeeded in restricting SIS to structured matrices while preserving a worst-case to average-case reduction. The worst-case problem is a restriction of a standard lattice problem to the specific family of cyclic lattices. The structure of Micciancio's matrices allows for an interpretation in terms of arithmetic in the ring $\mathbb{Z}_q[x]/(x^n - 1)$, where n is the dimension of the worst-case lattices and q is a small prime. Micciancio's construction leads to a family of pre-image resistant hash functions, with complexity quasi-linear in n. Peikert, Rosen, Lyubashevsky and Micciancio [30, 19] later suggested to change the ring to $\mathbb{Z}_q[x]/\Phi$ with a Φ that is irreducible over the rationals, sparse, and with small coefficients (e.g., $\Phi = x^n + 1$ for n a power of 2). The resulting hash function was proven collision-resistant under the assumed hardness of the modified average-case problem, called Ideal-SIS. The latter was itself proven at least as hard as the restrictions of standard worst-case lattice problems to a specific class of lattices (called ideal lattices). In 2009, Stehlé *et al.* [35] introduced a structured variant of LWE, which they proved as hard as Ideal-SIS (under a quantum reduction), and allowed for the design of an asymptotically efficient CPA-secure encryption scheme. In an independent concurrent work, Lyubashevsky *et al.* [21] proposed a ring variant of LWE, called R-LWE, whose great flexibility allows for more natural (and efficient) cryptographic constructions.

OUR RESULTS. The high efficiency and industrial standardization of NTRUEncrypt strongly motivate a theoretically founded study of its security. Indeed, in the absence of such a study so far, its security has remained in doubt over the last 15 years since its publication. In this paper, we address this problem. We prove that a mild modification of NTRUEncrypt is CPA-secure, assuming the quantum hardness of standard worst-case problems over ideal lattices (for $\Phi = x^n + 1$ with n a power of 2). The NTRUEncrypt modifications are summarized below. We stress

that our main goal in this paper is to provide a firm theoretical grounding for the security of `NTRUEncrypt` in the asymptotic sense. We leave to future work the consideration of practical issues, in particular the selection of concrete parameters for given security levels. As for other lattice-based schemes, the latter requires evaluation of security against practical lattice reduction attacks, which is out of the scope of the current work.

Our main contribution is the modification and analysis of the key generation algorithm. The secret key consists of two sparse polynomials of degrees $< n$ and coefficients of magnitude at most c, for a small constant c (typically, $c \in \{2, 3\}$). The public key is their quotient in $\mathbb{Z}_q[x]/(x^n - 1)$ (the denominator is resampled if it is not invertible). A simple information-theoretic argument shows that the public key cannot be uniformly distributed in the whole ring. It may be possible to extend the results of [4] to show that it is "well-spread" in the ring, but it still would not suffice for showing its cryptographic pseudorandomness, which seems necessary for exploiting the established hardness of R-LWE. To achieve a public key distribution statistically close to uniform, we sample the secret key polynomials according to a discrete Gaussian with standard deviation $\approx q^{1/2}$. An essential ingredient, which could be of independent interest, is a new regularity result for the ring $R_q := \mathbb{Z}_q[x]/(x^n + 1)$ when the polynomial $x^n + 1$ (with n a power of 2) has n factors modulo prime q: given $a_1, \ldots, a_m$ uniform in R_q, we would like $\sum_{i \leq m} s_i a_i$ to be within exponentially small statistical distance to uniformity, with small random s_i's and small m. Note that a similar regularity bound can be obtained with an FFT-based technique recently developed by Lyubashevsky, Peikert and Regev [22]. An additional difficulty in the public-key 'uniformity' proof, which we handle via an inclusion-exclusion argument, is that we need the s_i's to be invertible in R_q (the denominator of the public key is one such s_i): we thus sample according to a discrete Gaussian, and reject the sample if it is not invertible.

Brief comparison of `NTRUEncrypt` and its provably secure variant

Let R_{NTRU} be the ring $\mathbb{Z}[x]/(x^n - 1)$ with n prime. Let q be a medium-size integer (typically, either a prime or a power of 2 of the same order of magnitude as n). Finally, let $p \in R_{\text{NTRU}}$ with small coefficients, co-prime with q and such that the plaintext space R_{NTRU}/p is large (typically, one may take $p \in \{2, 3\}$ or $p = x + 2$).

The `NTRUEncrypt` secret key is a pair of polynomials $(f, g) \in R_{\text{NTRU}}^2$ that are sampled randomly with large prescribed proportions of zeros, and with their other coefficients having small magnitude. For improved decryption efficiency, one may choose f such that $f = 1 \bmod p$ (a typical choice [17] is to choose g and F with coefficients in $\{0, 1\}$ and set $f = 1 + p \cdot F$). With high probability, the polynomial f is invertible modulo q and modulo p, and if that is the case, the public-key is $h = pg/f \bmod q$ (otherwise, the key generation process is restarted). To encrypt a message $M \in R_{\text{NTRU}}/p$, one samples a random element $s \in R_{\text{NTRU}}$ of small Euclidean norm and computes the ciphertext $C = hs + M \bmod q$. The following procedure allows the owner of the secret key to decrypt:

- Compute $fC \bmod q$. If C was properly generated, this gives $pgs + fM \bmod q$. Since p, g, s, f, M have small coefficients, it can be expected that after reduction modulo q the obtained representative is $pgs + fM$ (in R_{NTRU}).
- Reduce the latter modulo p. This should provide $fM \bmod p$.
- Multiply the result of the previous step by the inverse of f modulo p (this step becomes vacuous if $f = 1 \bmod p$).

Note that the encryption process is probabilistic, and that decryption errors can occur for some sets of parameters. However, it is possible to arbitrarily decrease the decryption error probability, and even to eliminate it completely.

In order to achieve CPA-security we make a few modifications to the original `NTRUEncrypt` (which preserve its quasi-linear time and space complexity):

1. We replace R_{NTRU} by $R = \mathbb{Z}[x]/(x^n + 1)$ with n a power of 2. We will exploit the irreducibility of $x^n + 1$ and the fact that R is the ring of integers of a cyclotomic number field.
2. We choose a prime $q \leq \mathcal{P}oly(n)$ such that $f = x^n + 1 \bmod q$ has n distinct linear factors (i.e., $q = 1 \bmod 2n$). This allows us to use the search to decision reduction for R-LWE with ring $R_q := R/q$ (see [21]), and also to take $p = 2$.
3. We sample f and g from discrete Gaussians over R, rejecting the samples that are not invertible in R_q. We show that $f/g \bmod q$ is essentially uniformly distributed over the set of invertible elements of R_q. We may also choose $f = pf' + 1$ with f' sampled from a discrete Gaussian, to simplify decryption.
4. We add a small error term e in the encryption: $C = hs + pe + M \bmod q$, with s and e sampled from the R-LWE error distribution. This allows us to derive CPA security from the hardness of a variant of R-LWE (which is similar to the variant of LWE from [3, Se. 3.1]).

Work in progress and open problems

Our study is restricted to the sequence of rings $\mathbb{Z}[x]/\Phi_n$ with $\Phi_n = x^n + 1$ with n a power of 2. An obvious drawback is that this does not allow for much flexibility on the choice of n (in the case of NTRU, the degree was assumed prime, which provides more freedom). The R-LWE problem is known to be hard when Φ_n is cyclotomic [21]. We chose to restrict ourselves to cyclotomic polynomials of order a power of 2 because it makes the error generation of R-LWE more efficient, and the description of the schemes simpler to follow. Our results are likely to hold for more general rings than those we considered. An interesting choice could be the cyclotomic rings of prime order (i.e., $\Phi_n = (x^n - 1)/(x - 1)$ with n prime) as these are large subrings of the NTRU rings (and one might then be able to show that the hardness carries over to the NTRU rings).

An interesting open problem is to obtain a CCA secure variant of our scheme in the standard model, while maintaining its efficiency (within constant factors). The selection of concrete parameters based on practical security estimates for the worst-case SVP in ideal lattices or the average-case hardness of R-LWE/Ideal-SIS is also left as a future work.

The authors of NTRUEncrypt also proposed a signature scheme based on a similar design. The history of NTRUSign started with NSS in 2001 [16]. Its development has been significantly more hectic and controversial, with a series of cryptanalyses and repairs (see the survey [13]). In a work in progress, we construct a variant of NTRUSign with unforgeability related to the worst-case hardness of standard problems over ideal lattices, in the random oracle model. Our construction modifies the NTRUSign key generation and adapts the GPV signature scheme [11] to this setting.

Like NTRUEncrypt, Gentry's somewhat homomorphic scheme [9] also has ciphertexts consisting of a single ring element. It also admits a security proof under the assumed quantum hardness of standard worst-case problems over ideal lattices [10]. Our security analysis for the modified NTRUEncrypt scheme allows encrypting and decrypting $\Omega(n)$ plaintext bits for $\widetilde{O}(n)$ bit operations, while achieving security against $2^{g(n)}$-time attacks, for any $g(n)$ that is $\Omega(\log n)$ and $o(n)$, assuming the worst-case hardness of $\mathcal{P}oly(n)$-Ideal-SVP against $2^{O(g(n))}$-time quantum algorithms. The latter assumption is believed to be valid for any $g(n) = o(n)$. Gentry's analysis from [10, 8] can be generalized to handle $2^{g(n)}$-time attacks while encrypting and decrypting $O(g(n))$ plaintext bits for $\widetilde{O}(n)$ bit operations, under the assumed hardness of $2^{\Omega(g(n))}$-Ideal-SVP against $2^{O(g(n))}$-time quantum algorithms. The latter assumption is known to be invalid when $g(n) = \widetilde{\Omega}(\sqrt{n})$ (using [34]), thus limiting the attacker's strength the analysis can handle. On the other hand, Gentry's scheme allows homomorphic additions and multiplications, whereas ours seems restricted to additions. Our scheme and Gentry's seem to be closely related, and we leave to future work the further investigation of this relation.

NOTATION. We denote by $\rho_\sigma(\boldsymbol{x})$ (resp. ν_σ) the standard n-dimensional Gaussian function (resp. distribution) with center $\mathbf{0}$ and variance σ, i.e., $\rho_\sigma(\boldsymbol{x}) = \exp(-\pi\|\boldsymbol{x}\|^2/\sigma^2)$ (resp. $\nu_\sigma(\boldsymbol{x}) = \rho_\sigma(\boldsymbol{x})/\sigma^n$). We denote by $\mathrm{Exp}(\mu)$ the exponential distribution on $\mathbb{R}$ with mean μ and by $U(E)$ the uniform distribution over a finite set E . If D_1 and D_2 are two distributions on discrete domain E, their statistical distance is $\Delta(D_1; D_2) = \frac{1}{2}\sum_{x\in E} |D_1(x) - D_2(x)|$. We write $z \hookleftarrow D$ when the random variable z is sampled from the distribution D.

2 A Few Background Results

A (full-rank) *lattice* is a set of the form $L = \sum_{i\leq n} \mathbb{Z}\boldsymbol{b}_i$, where the $\boldsymbol{b}_i$'s are linearly independent vectors in $\mathbb{R}^n$. The integer n is called the *lattice dimension*, and the $\boldsymbol{b}_i$'s are called a *basis* of L. The *minimum* $\lambda_1(L)$ (resp. $\lambda_1^\infty(L)$) is the Euclidean (resp. infinity) norm of any shortest vector of $L \setminus \mathbf{0}$. If $B = (\boldsymbol{b}_i)_i$ is a basis matrix of L, the *fundamental parallelepiped* of B is the set $\mathcal{P}(B) = \{\sum_{i\leq n} c_i\boldsymbol{b}_i : c_i \in [0, 1)\}$. The volume $|\det B|$ of $\mathcal{P}(B)$ is an invariant of the lattice L which we denote by $\det L$. Minkowski's theorem states that $\lambda_1(L) \leq \sqrt{n}(\det L)^{1/n}$. More generally, the k-th *minimum* $\lambda_k(L)$ for $k \leq n$ is defined as the smallest r such that L contains $\geq k$ linearly independent vectors of norm $\leq r$. The *dual* of L is the lattice $\widehat{L} = \{\boldsymbol{c} \in \mathbb{R}^n : \forall i, \langle \boldsymbol{c}, \boldsymbol{b}_i\rangle \in \mathbb{Z}\}$.

For a lattice $L \subseteq \mathbb{R}^n$, $\sigma > 0$ and $\boldsymbol{c} \in \mathbb{R}^n$, we define the *lattice Gaussian distribution* of support L, deviation σ and center $\boldsymbol{c}$ by $D_{L,\sigma,\boldsymbol{c}}(\boldsymbol{b}) = \frac{\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{b})}{\rho_{\sigma,\boldsymbol{c}}(L)}$, for any $\boldsymbol{b} \in L$. We will omit the subscript $\boldsymbol{c}$ when it is $\mathbf{0}$. We extend the definition of $D_{L,\sigma,\boldsymbol{c}}$ to any $M \subseteq L$ (not necessarily a sublattice), by setting $D_{M,\sigma,\boldsymbol{c}}(\boldsymbol{b}) = \frac{\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{b})}{\rho_{\sigma,\boldsymbol{c}}(M)}$. For $\delta > 0$, we define the *smoothing parameter* $\eta_\delta(L)$ as the smallest $\sigma > 0$ such that $\rho_{1/\sigma}(\widehat{L} \setminus \mathbf{0}) \leq \delta$. It quantifies how large σ needs to be for $D_{L,\sigma,\boldsymbol{c}}$ to behave like a continuous Gaussian. We will typically consider $\delta = 2^{-n}$.

Lemma 1 ([24, Le. 3.3]). *For any full-rank lattice $L \subseteq \mathbb{R}^n$ and $\delta \in (0,1)$, we have $\eta_\delta(L) \leq \sqrt{\ln(2n(1+1/\delta))/\pi} \cdot \lambda_n(L)$.*

Lemma 2 ([28, Le. 3.5]). *For any full-rank lattice $L \subseteq \mathbb{R}^n$ and $\delta \in (0,1)$, we have $\eta_\delta(L) \leq \sqrt{\ln(2n(1+1/\delta))/\pi}/\lambda_1^\infty(\widehat{L})$.*

Lemma 3 ([24, Le. 4.4]). *For any full-rank lattice $L \subseteq \mathbb{R}^n$, $\boldsymbol{c} \in \mathbb{R}^n$, $\delta \in (0,1)$ and $\sigma \geq \eta_\delta(L)$, we have $\Pr_{\boldsymbol{b} \hookleftarrow D_{L,\sigma,\boldsymbol{c}}}[\|\boldsymbol{b}\| \geq \sigma\sqrt{n}] \leq \frac{1+\delta}{1-\delta} 2^{-n}$.*

Lemma 4 ([11, Cor. 2.8]). *Let $L' \subseteq L \subseteq \mathbb{R}^n$ be full-rank lattices. For any $\boldsymbol{c} \in \mathbb{R}^n$, $\delta \in (0, 1/2)$ and $\sigma \geq \eta_\delta(L')$, we have $\Delta(D_{L,\sigma,\boldsymbol{c}} \bmod L'; U(L/L')) \leq 2\delta$.*

Lemma 5 ([11, Th. 4.1]). *There exists a polynomial-time algorithm that takes as input any basis $(\boldsymbol{b}_i)_i$ of any lattice $L \subseteq \mathbb{Z}^n$ and $\sigma = \omega(\sqrt{\log n}) \max \|\boldsymbol{b}_i\|$ (resp. $\sigma = \Omega(\sqrt{n}) \max \|\boldsymbol{b}_i\|$), and returns samples from a distribution whose statistical distance to $D_{L,\sigma}$ is negligible (resp. exponentially small) with respect to n.*

The most famous lattice problem is SVP. Given a basis of a lattice L, it aims at finding a shortest vector in $L \setminus \mathbf{0}$. It can be relaxed to γ-SVP by asking for a non-zero vector that is no longer than $\gamma(n)$ times a solution to SVP, for a prescribed function $\gamma(\cdot)$. It is believed that no subexponential quantum algorithm solves the computational variants of γ-SVP in the worst case, for any $\gamma \leq \mathcal{P}oly(n)$. The smallest γ which is known to be achievable in polynomial time is exponential, up to poly-logarithmic factors in the exponent ([34, 26]).

Ideal lattices and algebraic number theory

IDEAL LATTICES. Let n a power of 2 and $\Phi = x^n + 1$ (which is irreducible over $\mathbb{Q}$). Let R be the ring $\mathbb{Z}[x]/\Phi$. An (integral) ideal I of R is a subset of R closed under addition and multiplication by arbitrary elements of R. By mapping polynomials to the vectors of their coefficients, we see that an ideal $I \neq 0$ corresponds to a full-rank sublattice of $\mathbb{Z}^n$: we can thus view I as both a lattice and an ideal. An *ideal lattice* for Φ is a sublattice of $\mathbb{Z}^n$ that corresponds to a non-zero ideal $I \subseteq R$. The *algebraic norm* $\mathcal{N}(I)$ is the cardinality of the additive group R/I. It is equal to $\det I$, where I is regarded as a lattice. Any non-zero ideal I of R satisfies $\lambda_n(I) = \lambda_1(I)$. In the following, an ideal lattice will implicitly refer to a Φ-ideal lattice.

By restricting SVP (resp. γ-SVP) to instances that are ideal lattices, we obtain Ideal-SVP (resp. γ-Ideal-SVP). The latter is implicitly parameterized by

the sequence of polynomials $\Phi_n = x^n + 1$, where n is restricted to powers of 2. No algorithm is known to perform non-negligibly better for (γ-)Ideal-SVP than for (γ-)SVP.

PROPERTIES OF THE RING R. For $v \in R$ we denote by $\|v\|$ its Euclidean norm (as a vector). We define the multiplicative *expansion factor* $\gamma_\times(R)$ by $\gamma_\times(R) = \max_{u,v \in R} \frac{\|u \times v\|}{\|u\| \cdot \|v\|}$. For our choice of Φ, we have $\gamma_\times(R) = \sqrt{n}$ (see [9, p. 174]).

Since Φ is the $2n$-th cyclotomic polynomial, the ring R is exactly the maximal order (i.e., the ring of integers) of the cyclotomic field $\mathbb{Q}[\zeta] \cong \mathbb{Q}[x]/\Phi =: K$, where $\zeta \in \mathbb{C}$ is a primitive $2n$-th root of unity. We denote by $(\sigma_i)_{i \leq n}$ the canonical complex embeddings: We can choose $\sigma_i : P \mapsto P(\zeta^{2i+1})$ for $i \leq n$. For any α in $\mathbb{Q}[\zeta]$, we define its T_2-norm by $T_2(\alpha)^2 = \sum_{i \leq n} |\sigma_i(\alpha)|^2$ and its algebraic norm by $\mathcal{N}(\alpha) = \prod_{i \leq n} |\sigma_i(\alpha)|$. The arithmetic-geometric inequality gives $\mathcal{N}(\alpha)^{2/n} \leq \frac{1}{n} T_2(\alpha)^2$. Also, for the particular cyclotomic fields we are considering, the polynomial norm (the norm of the coefficient vector of α when expressed as an element of K) satisfies $\|\alpha\| = \frac{1}{\sqrt{n}} T_2(\alpha)$. We also use the fact that for any $\alpha \in R$, we have $|\mathcal{N}(\alpha)| = \det \langle \alpha \rangle$, where $\langle \alpha \rangle$ is the ideal of R generated by α. For simplicity, we will try to use the polynomial terminology wherever possible.

Let q be a prime number such that Φ has n distinct linear factors modulo q (i.e., $q = 1 \bmod 2n$): $\Phi = \prod_{i \leq n} \Phi_i = \prod_{i \leq n} (x - \phi_i) \bmod q$. Let $R_q = R/qR = \mathbb{Z}_q[x]/\Phi$. Dirichlet's theorem on arithmetic progressions implies that infinitely such primes exist. Furthermore, Linnik's theorem asserts that the smallest such q is $\mathcal{P}oly(n)$, and much effort has been spent to decrease this bound (the current record seems to be $O(n^{5.2})$, see [36]). Furthermore, we can write ϕ_i as r^i, where r is a primitive $(2n)$-th root of unity modulo q. This implies that the Chinese Remainder Theorem in R_q provides a natural fast Discrete Fourier Transform, and thus multiplication of elements of R_q can be performed within $O(n \log n)$ additions and multiplications modulo q (see [7, Ch. 8], [20, Se. 2.1]).

The R-LWE problem

For $s \in R_q$ and ψ a distribution in R_q, we define $A_{s,\psi}$ as the distribution obtained by sampling the pair $(a, as+e)$ with $(a, e) \hookleftarrow U(R_q) \times \psi$. The Ring Learning With Errors problem (R-LWE) was introduced by Lyubashevsky *et al.* [21] and shown hard for specific error distributions ψ. These are slightly technical to define (see below), but for the present work, the important facts to be remembered are that the samples are small (see Lemma 6), and can be obtained in quasi-linear time.

The error distributions ψ that we use are an adaptation of those introduced in [21]. They are sampled from a family of distributions $\overline{\Upsilon}_\alpha$ that we now define. For $\boldsymbol{\sigma} \in \mathbb{R}^n$ with positive coordinates, we define the ellipsoidal Gaussian $\rho_{\boldsymbol{\sigma}}$ as the row vector of independent Gaussians $(\rho_{\sigma_1}, \ldots, \rho_{\sigma_n})$, where $\sigma_i = \sigma_{i+n/2}$ for $1 \leq i \leq n/2$. As we want to define R-LWE in the polynomial expression of R rather than with the so-called "space H" of [21], we apply a matrix transformation to the latter Gaussians. We define a sample from $\rho'_{\boldsymbol{\sigma}}$ as a sample

from $\rho_{\boldsymbol{\sigma}}$, multiplied first (from the right) by $\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ i & -i\end{pmatrix}\otimes \mathrm{Id}_{n/2} \in \mathbb{C}^{n\times n}$, and second by $V = \frac{1}{n}\left(\zeta^{-(2j+1)k}\right)_{0\le j,k<n}$. Note that vector multiplication by matrix V corresponds to a complex discrete Fourier transform, and can be performed in $O(n\log n)$ complex-valued arithmetic operations with the Cooley-Tukey FFT. Moreover, it is numerically extremely stable: if all operations are performed with a precision of $p = \Omega(\log n)$ bits, then the computed output vector $fl(\boldsymbol{y})$ satisfies $\|fl(\boldsymbol{y})-\boldsymbol{y}\| \le C\cdot(\log n)\cdot 2^{-p}\cdot\|\boldsymbol{y}\|$, where C is some absolute constant and $\boldsymbol{y}$ is the vector that would be obtained with exact computations. We refer to [12, Se. 24.1] for details. We now define a sample from $\overline{\rho'}_{\boldsymbol{\sigma}}$ as follows: compute a sample from $\rho'_{\boldsymbol{\sigma}}$ with absolute error $< 1/n^2$; if it is within distance $1/n^2$ of the middle of two consecutive integers, then restart; otherwise, round it to a closest integer and then reduce it modulo q. Finally, a distribution sampled from $\overline{\Upsilon}_\alpha$ for $\alpha \ge 0$ is defined as $\overline{\rho'}_{\boldsymbol{\sigma}}$, where $\sigma_i = \sqrt{\alpha^2q^2 + x_i}$ with the x_i's sampled independently from the distribution $\mathrm{Exp}(n\alpha^2q^2)$.

Sampling from $\rho'_{\boldsymbol{\sigma}}$ can be performed in time $\widetilde{O}(n)$. Sampling from $\overline{\Upsilon}_\alpha$ can also be performed in expected time $\widetilde{O}(n)$, and the running-time is bounded by a quantity that follows a geometric law of parameter < 1. Furthermore, in all our cryptographic applications, one could pre-compute such samples off-line (i.e., before the message M to be processed is known).

Lemma 6. *Assume that $\alpha q \ge \sqrt{n}$. For any $r \in R$, we have $\Pr_{y\hookleftarrow\overline{\Upsilon}_\alpha}[\|yr\|_\infty \ge \alpha q\omega(\log n)\cdot\|r\|] \le n^{-\omega(1)}$.*

Proof. We define Υ_α exactly as $\overline{\Upsilon}_\alpha$, but without the rejection step from $\rho'_{\boldsymbol{\sigma}}$ to $\overline{\rho'}_{\sigma}$. Because of the bound on the rejection probability, it suffices to prove the result with Υ_α instead of $\overline{\Upsilon}_\alpha$. Let $y \hookleftarrow \Upsilon_\alpha$. The involved $\boldsymbol{\sigma}$ satisfies $\sigma_k = \sqrt{\alpha^2q^2 + x_k}$, with the x_k's sampled independently from the distribution $\mathrm{Exp}(n\alpha^2q^2)$. We have $\max\sigma_k \le \alpha q\sqrt{n}\omega(\sqrt{\log n})$ with probability $1-n^{-\omega(1)}$. We write $y = y'+\eta$, where the field element $y' \in K$ is sampled from $\rho'_{\boldsymbol{\sigma}}$, and actually derived from a sample $\boldsymbol{z}$ from $\rho_{\boldsymbol{\sigma}}$, and $\eta \in K$ is the error in rounding $y' \in K$ to $y \in R$, with $\|\eta\|_\infty \le 1/2$. Then $\|yr\|_\infty \le \|y'r\|_\infty + \|\eta r\|_\infty$. Using the Schwartz inequality, the second term can be bounded as $\|\eta r\|_\infty \le \frac{\sqrt{n}}{2}\|r\|$. We now bound the first term. The embedding vector of y' has the following shape:

$$\frac{1}{\sqrt{2}}(z_1 + iz_{n/2+1}, \ldots, z_{n/2} + iz_n, z_1 - iz_{n/2+1}, \ldots, z_{n/2} - iz_n).$$

Let $(r^{(k)})_k$ be the embedding vector of r. Then the embedding vector of $y'r$ is $(y'^{(k)}r^{(k)})_k$. The coefficient in x^j of $y'r$ is

$$\begin{aligned}\frac{1}{n}\sum_{0\le k<n}\zeta^{-(2j+1)k}y^{(k)}r^{(k)} &= \frac{2}{n}\Re\left(\sum_{0\le k<n/2}\zeta^{-(2j+1)k}y^{(k)}r^{(k)}\right)\\ &= \frac{\sqrt{2}}{n}\sum_{0\le k<n/2}\Re\left((\zeta^{-(2j+1)k}r^{(k)})(z_{k+1} + iz_{n/2+k+1})\right).\end{aligned}$$

The kth summand of the last sum follows a normal law of mean 0 and standard deviation $|r^{(k)}|\sigma_k$. Therefore, the coefficient in x^j of yr follows a normal law of standard deviation $\leq \frac{1}{n} T_2(r) \max \sigma_k$, which is $\leq \frac{1}{\sqrt{n}} \alpha q \omega(\sqrt{\log n}) \cdot T_2(r) = \alpha q \omega(\log n) \cdot \|r\|$ with probability $1 - n^{-\omega(1)}$. Using $\alpha q \geq \sqrt{n}$, we get $\|yr\|_\infty + \|\eta r\|_\infty \leq \alpha q \omega(\log n) \cdot \|r\|$ with probability $1 - n^{-\omega(1)}$, as claimed. □

We now define our adaptation of R-LWE.

Definition 1. *The Ring Learning With Errors Problem with parameters q, α and Φ (R-LWE$^{\Phi}_{q,\alpha}$) is as follows. Let $\psi \hookleftarrow \overline{\Upsilon}_\alpha$ and $s \hookleftarrow U(R_q)$. Given access to an oracle $\mathcal{O}$ that produces samples in $R_q \times R_q$, distinguish whether $\mathcal{O}$ outputs samples from $A_{s,\psi}$ or from $U(R_q \times R_q)$. The distinguishing advantage should be $1/\mathcal{P}oly(n)$ (resp. $2^{-o(n)}$) over the randomness of the input, the randomness of the samples and the internal randomness of the algorithm.*

The following theorem indicates that R-LWE is hard, assuming that the worst-case γ-Ideal-SVP cannot be efficiently solved using quantum computers, for small γ. It was recently improved by Lyubashevsky *et al.* [22]: if the number of samples that can be asked to the oracle $\mathcal{O}$ is bounded by a constant (which is the case in our application), then the result also holds with simpler errors than $e \hookleftarrow \psi \hookleftarrow \overline{\Upsilon}_\alpha$, and with an even smaller Ideal-SVP approximation factor γ. This should allow to both simplify the modified `NTRUEncrypt` and to strengthen its security guarantee.

Theorem 1 (Adapted from [21]). *Assume that $\alpha q = \omega(n\sqrt{\log n})$ (resp. $\Omega(n^{1.5})$) with $\alpha \in (0,1)$ and $q = \mathcal{P}oly(n)$. There exists a randomized polynomial-time (resp. subexponential) quantum reduction from γ-Ideal-SVP to R-LWE$_{q,\alpha}$, with $\gamma = \omega(n^{1.5} \log n)/\alpha$ (resp. $\Omega(n^{2.5})/\alpha$).*

The differences with [21] in the above formulation are the use of the polynomial representation (handled by applying the complex FFT to the error term), the use of R_q rather than $R_q^\vee := R^\vee/q$ where $R^\vee$ is the codifferent (here we have $R_q^\vee = \frac{1}{n} R_q$), and the truncation of the error to closest integer if it is far from the middle of two consecutive integers. The new variant remains hard because a sample passes the rejection step with non-negligible probability, and the rounding can be performed on the oracle samples obliviously to the actual error.

VARIANTS OF R-LWE. For $s \in R_q$ and ψ a distribution in R_q, we define $A^\times_{s,\psi}$ as the distribution obtained by sampling the pair $(a, as + e)$ with $(a, e) \hookleftarrow U(R_q^\times) \times \psi$, where $R_q^\times$ is the set of invertible elements of R_q. When $q = \Omega(n)$, the probability for a uniform element of R_q of being invertible is non-negligible, and thus R-LWE remains hard even when $A_{s,\psi}$ and $U(R_q \times R_q)$ are respectively replaced by $A^\times_{s,\psi}$ and $U(R_q^\times \times R_q)$. We call R-LWE$^\times$ the latter variant.

Furthermore, similarly to [3, Le. 2] and as explained in [22], the nonce s can also be chosen from the error distribution without incurring any security loss. We call R-LWE$^\times_{\text{HNF}}$ the corresponding modification of R-LWE. We recall the argument, for completeness. Assume an algorithm $\mathcal{A}$ can solve R-LWE$^\times_{\text{HNF}}$. We use $\mathcal{A}$ to solve R-LWE$^\times$. The principle is to transform samples $((a_i, b_i))_i$

into samples $((a_1^{-1}a_i, b_i - a_1^{-1}b_1 a_i))_i$, where inversion is performed in $R_q^\times$. This transformation maps $A^\times_{s,\psi}$ to $A^\times_{-e_1,\psi}$, and $U(R_q^\times \times R_q)$ to itself.

3 New Results on Module q-ary Lattices

In this section, we present strong regularity bounds for the ring R_q. For this purpose, we first study two families of R-modules.

3.1 Duality results for some module lattices

Let $\boldsymbol{a} \in R_q^m$. We define the following families of R-modules, for I an arbitrary ideal of R_q:

$$\boldsymbol{a}^\perp(I) := \{(t_1, \ldots, t_m) \in R^m : \forall i, (t_i \bmod q) \in I \text{ and } \sum_i t_i a_i = 0 \bmod q\},$$

$$L(\boldsymbol{a}, I) := \{(t_1, \ldots, t_m) \in R^m : \exists s \in R_q, \forall i, (t_i \bmod q) = a_i \cdot s \bmod I\}.$$

We also define $\boldsymbol{a}^\perp$ and $L(\boldsymbol{a})$ as $\boldsymbol{a}^\perp(R_q)$ and $L(\boldsymbol{a}, \langle 0 \rangle)$ respectively. The ideals of R_q are of the form $I_S := \prod_{i \in S}(x - \phi_i) \cdot R_q = \{a \in R_q : \forall i \in S, a(\phi_i) = 0\}$, where S is any subset of $\{1, \ldots, n\}$ (the ϕ_i's are the roots of Φ modulo q). We define $I_S^\times = \prod_{i \in S}(x - \phi_i^{-1}) \cdot R_q$.

Lemma 7. *Let $S \subseteq \{1, \ldots, n\}$ and $\boldsymbol{a} \in R_q^m$. Let $\overline{S} = \{1, \ldots, n\} \setminus S$ and $\boldsymbol{a}^\times \in R_q^m$ be defined by $a_i^\times = a_i(x^{-1})$. Then (considering both sets as mn-dimensional lattices by identifying R and $\mathbb{Z}^n$):*

$$\widehat{\boldsymbol{a}^\perp(I_S)} = \frac{1}{q} L(\boldsymbol{a}^\times, I_{\overline{S}}^\times).$$

Proof. We first prove that $\frac{1}{q} L(\boldsymbol{a}^*, I_{\overline{S}}^\times) \subseteq \widehat{\boldsymbol{a}^\perp(I_S)}$. Let $(t_1, \ldots, t_m) \in \boldsymbol{a}^\perp(I_S)$ and $(t'_1, \ldots, t'_m) \in L(\boldsymbol{a}^*, I_{\overline{S}})$. Write $t_i = \sum_{j<n} t_{i,j} x^j$ and $t'_i = \sum_{j<n} t'_{i,j} x^j$ for any $i \leq m$. Our goal is to show that $\sum_{i \leq m, j \leq n} t_{i,j} t'_{i,j} = 0 \bmod q$. This is equivalent to showing that the constant coefficient of the polynomial $\sum_{i \leq m} t_i(x) t'_i(x^{-1})$ is 0 modulo q. It thus suffices to show that $\sum_{i \leq m} t_i(x) t'_i(x^{-1}) \bmod q = 0$ (in R_q). By definition of the t'_i's, there exists $s \in R_q$ such that $(t'_i \bmod q) = a_i^\times \cdot s + b'_i$ for some $b'_i \in I_{\overline{S}}^\times$. We have the following, modulo q:

$$\sum_{i \leq m} t_i(x) t'_i(x^{-1}) = s(x^{-1}) \cdot \sum_{i \leq m} t_i(x) a_i(x) + \sum_{i \leq m} t_i(x) b'_i(x^{-1}).$$

Both sums in the right hand side evaluate to 0 in R_q, which provides the desired inclusion.

To complete the proof, it suffices to show that $\widehat{L(\boldsymbol{a}^\times, I_{\overline{S}}^\times)} \subseteq \frac{1}{q} \boldsymbol{a}^\perp(I_S)$. It can be seen by considering the elements of $L(\boldsymbol{a}^\times, I_{\overline{S}})$ corresponding to $s = 1$. □

3.2 On the absence of unusually short vectors in $L(\boldsymbol{a}, I_S)$

We show that for $\boldsymbol{a} \hookleftarrow U((R_q^\times)^m)$, the lattice $L(\boldsymbol{a}, I_S)$ is extremely unlikely to contain unusually short vectors for the infinity norm, i.e., much shorter than guaranteed by the Minkowski upper bound $\det(L(\boldsymbol{a}, I_S))^{\frac{1}{mn}} = q^{(1-\frac{1}{m})\frac{|S|}{n}}$ (we have $\det(L(\boldsymbol{a}, I_S)) = q^{(m-1)|S|}$ because there are $q^{n+(m-1)(n-|S|)}$ points of $L(\boldsymbol{a}, I_S)$ in the cube $[0, q-1]^{mn}$). Note that our lower bound approaches the Minkowski bound as $\frac{|S|}{n}$ approaches 1, but becomes progressively looser as $\frac{|S|}{n}$ drops towards $\approx 1 - \frac{1}{m}$. Fortunately, for our applications, we will be using this bound with $\frac{|S|}{n} = 1 - \varepsilon$ for some small ε, where the bound is close to being tight.

Lemma 8. *Let $n \geq 8$ be a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q \geq 5$. Then, for any $S \subseteq \{1, \ldots, n\}$, $m \geq 2$ and $\varepsilon > 0$, we have $\lambda_1^\infty(L(\boldsymbol{a}, I_S)) \geq \frac{1}{\sqrt{n}} q^\beta$, with:*

$$\begin{aligned}\beta &:= 1 - \frac{1}{m} + \frac{1 - \sqrt{1 + 4m(m-1)\left(1 - \frac{|S|}{n}\right) + 4m\varepsilon}}{2m} \\ &\geq 1 - \frac{1}{m} - \varepsilon - (m-1)\left(1 - \frac{|S|}{n}\right),\end{aligned}$$

except with probability $\leq 2^n (q-1)^{-\varepsilon n}$ over the uniformly random choice of $\boldsymbol{a}$ in $(R_q^\times)^m$.

Proof. Recall that $\Phi = \prod_{i \leq n} \Phi_i$ for distinct linear factors Φ_i. By the Chinese Remainder Theorem, we know that R_q (resp. $R_q^\times$) is isomorphic to $(\mathbb{Z}_q)^n$ (resp. $(\mathbb{Z}_q^\times)^n$) via the isomorphism $t \mapsto (t \bmod \Phi_i)_{i \leq m}$. Let $g_{I_S} = \prod_{i \in S} \Phi_i$: it is a degree $|S|$ generator of I_S.

Let p denote the probability (over the randomness of $\boldsymbol{a}$) that $L(\boldsymbol{a}, I_S)$ contains a non-zero vector $\boldsymbol{t}$ of infinity norm $< B$, where $B = \frac{1}{\sqrt{n}} q^\beta$. We upper bound p by the union bound, summing the probabilities $p(\boldsymbol{t}, s) = \Pr_{\boldsymbol{a}}[\forall i, t_i = a_i s \bmod I_S]$ over all possible values for $\boldsymbol{t}$ of infinity norm $< B$ and $s \in R_q / I_S$. Since the a_i's are independent, we have $p(\boldsymbol{t}, s) = \prod_{i \leq m} p_i(t_i, s)$, where $p_i(t_i, s) = \Pr_{a_i}[t_i = a_i s \bmod I_S]$.

Wlog we can assume that $\gcd(s, g_{I_S}) = \gcd(t_i, g_{I_S})$ (up to multiplication by an element of $\mathbb{Z}_q^\times$): If this is not the case, there exists $j \leq n$ such that either $t_i \bmod \Phi_j = 0$ and $s \bmod \Phi_j \neq 0$, or $t_i \bmod \Phi_j \neq 0$ and $s \bmod \Phi_j = 0$; In both cases, we have $p_i(t_i, s) = 0$ because $a_i \in R_q^\times$. We now assume that $\gcd(s, g_{I_S}) = \gcd(t_i, g_{I_S}) = \prod_{i \in S'} \Phi_i$ for some $S' \subseteq S$ of size $0 \leq d \leq |S|$. For any $j \in S'$, we have $t_i = a_i s = 0 \bmod \Phi_j$ regardless of the value of $a_i \bmod \Phi_j$, while for $j \in S \setminus S'$, we have $s \neq 0 \bmod \Phi_j$ and there exists a unique value of $a_i \bmod \Phi_j$ such that $t_i = a_i s \bmod \Phi_j$. Moreover for any $j \notin S$, the value of $a_i \bmod \Phi_j$ can be arbitrary in $\mathbb{Z}_q^\times$. So, overall, there are $(q-1)^{d+n-|S|}$ differents a_i's in $R_q^\times$ such that $t_i = a_i s \bmod I_S$. This leads to $p_i(t_i, s) = (q-1)^{d-|S|}$.

So far, we have showed that the probability p can be upper bounded by:

$$p \leq \sum_{0 \leq d \leq |S|} \sum_{\substack{h = \prod_{i \in S'} \Phi_i \\ S' \subseteq S \\ |S'| = d}} \sum_{\substack{s \in R_q/I_S \\ h|s}} \sum_{\substack{\boldsymbol{t} \in (R_q)^m \\ \forall i, 0 < \|t_i\|_\infty < B \\ \forall i, h|t_i}} \prod_{i \leq m} (q-1)^{d-|S|}.$$

For $h = \prod_{i \in S'} \Phi_i$ of degree d, let $N(B,d)$ denote the number of $t \in R_q$ such that $\|t\|_\infty < B$ and $t = ht'$ for some $t' \in R_q$ of degree $< n - d$. We consider two bounds for $N(B,d)$ depending on d.

Suppose that $d \geq \beta \cdot n$. Then we claim that $N(B,d) = 0$. Indeed, any $t = ht'$ for some $t' \in R_q$ belongs to the ideal $\langle h, q \rangle$ of R generated by h and q. For any non-zero $t \in \langle h, q \rangle$, we have $\mathcal{N}(t) = \mathcal{N}(\langle t \rangle) \geq \mathcal{N}(\langle h, q \rangle) = q^d$, where the inequality is because the ideal $\langle t \rangle$ is a full-rank sub-ideal of $\langle h, q \rangle$, and the last equality is because $\deg h = d$. It follows from the arithmetic-geometric inequality that $\|t\| = \frac{1}{\sqrt{n}} T_2(t) \geq \mathcal{N}(t)^{1/n} \geq q^{d/n}$. By equivalence of norms, we conclude that $\|t\|_\infty \geq \lambda_1^\infty(\langle h, q \rangle) \geq \frac{1}{\sqrt{n}} q^{d/n}$. We see that $d/n \geq \beta$ implies that $\|t\|_\infty \geq B$, so that $N(B,d) = 0$.

Suppose now that $d < \beta \cdot n$. Then we claim that $N(B,d) \leq (2B)^{n-d}$. Indeed, since the degree of h is d, the vector $\bar{t}$ formed by the $n-d$ low-order coefficients of t is related to the vector $\overline{t'}$ formed by the $n-d$ low-order coefficients of t' by a lower triangular $(n-d) \times (n-d)$ matrix whose diagonal coefficients are equal to 1. Hence this matrix is non-singular modulo q so the mapping from $\overline{t'}$ to $\bar{t}$ is one-to-one. This provides the claim.

Using the above bounds on $N(B,d)$, the fact that the number of subsets of S of cardinality d is $\leq 2^d$, and the fact that the number of $s \in R_q/I_S$ divisible by $h = \prod_{i \in S'} \Phi_i$ is $q^{|S|-d}$, the above bound on p implies

$$p \leq 2^n \max_{d \leq \beta \cdot n} \frac{(2B)^{m(n-d)}}{(q-1)^{(m-1)(|S|-d)}}.$$

With our choice of B, we have $2B \leq (q-1)^\beta$ (this is implied by $n \geq 8, q \geq 5$ and $\beta \leq 1$). A straightforward computation then leads to the claimed upper bound on p. □

3.3 Improved regularity bounds

We now study the uniformity of distribution of $(m+1)$-tuples from $(R_q^\times)^m \times R_q$ of the form $(a_1, \ldots, a_m, \sum_{i \leq m} t_i a_i)$, where the a_i's are independent and uniformly random in $R_q^\times$, and the t_i's are chosen from some distribution on R_q concentrated on elements with small coefficients. Similarly to [23], we call the distance of the latter distribution to the uniform distribution on $(R_q^\times)^m \times R_q$ the *regularity* of the generalized knapsack function $(t_i)_{i \leq m} \mapsto \sum_{i \leq m} t_i a_i$. For our NTRU application we are particularly interested in the case where $m = 2$.

The regularity result in [23, Se. 4.1] applies when the a_i's are uniformly random in the whole ring R_q, and the t_i's are uniformly random on the subset

of elements of R_q with coefficients of magnitude $\leq d$ for some $d < q$. In this case, the regularity bound from [23] is $\Omega(\sqrt{nq/d^m})$. Unfortunately, this bound is non-negligible for small m and q, e.g., for $m = O(1)$ and $q = \mathcal{P}oly(n)$. To make it exponentially small in n, one needs to set $m \log d = \Omega(n)$, which inevitably leads to inefficient cryptographic functions. When the a_i's are chosen uniformly from the whole ring R_q, the actual regularity is not much better than this undesirable regularity bound. This is because R_q contains n proper ideals of size $q^{n-1} = |R_q|/q$, and the probability $\approx n/q^m$ that all of the a_i's fall into one such ideal (which causes $\sum t_i a_i$ to also be trapped in the proper ideal) is non-negligible for small m. To circumvent this problem, we restrict the a_i's to be uniform in $R_q^\times$, and we choose the t_i's from a discrete Gaussian distribution. We show a regularity bound exponentially small in n even for $m = O(1)$, by using an argument similar to that used in [11, Se. 5.1] for unstructured generalized knapsacks, based on the *smoothing parameter* of the underlying lattices. Note that the new regularity result can be used within the Ideal-SIS trapdoor generation of [35, Se. 3], thus extending the latter to a fully splitting q. It also shows that the encryption scheme from [21] can be shown secure against subexponential (quantum) attackers, assuming the subexponential (quantum) hardness of standard worst-case problems over ideal lattices.

Theorem 2. *Let $n \geq 8$ be a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q \geq 5$. Let $m \geq 2$, $\varepsilon > 0$, $\delta \in (0, 1/2)$ and $\boldsymbol{t} \hookleftarrow D_{\mathbb{Z}^{mn},\sigma}$, with $\sigma \geq \sqrt{n \ln(2mn(1+1/\delta))/\pi} \cdot q^{\frac{1}{m}+\varepsilon}$. Then for all except a fraction $\leq 2^n(q-1)^{-\varepsilon n}$ of $\boldsymbol{a} \in (R_q^\times)^m$, we have $\eta_\delta(\boldsymbol{a}^\perp) \leq \sqrt{n \ln(2mn(1+1/\delta))/\pi} \cdot q^{\frac{1}{m}+\varepsilon}$, and the distance to uniformity of $\sum_{i \leq m} t_i a_i$ is $\leq 2\delta$. As a consequence:*

$$\Delta\left[\left(a_1, \ldots, a_m, \sum_{i \leq m} t_i a_i\right);\ U\left((R_q^\times)^m \times R_q\right)\right] \leq 2\delta + 2^n(q-1)^{-\varepsilon n}.$$

When using this result, one is typically interested in taking a small constant $\varepsilon > 0$, because it allows to lower the standard deviation σ and thus the required amount of randomness. Then a tiny δ should be chosen (e.g., $\delta \approx 2^n(q-1)^{-\varepsilon n}$), as it drastically lowers the statistical distance upper bound, without strengthening the standard deviation requirement much.

For each $\boldsymbol{a} \in (R_q^\times)^m$, let $D_{\boldsymbol{a}}$ denote the distribution of $\sum_{i \leq m} t_i a_i$ where $\boldsymbol{t}$ is sampled from $D_{\mathbb{Z}^{mn},\sigma}$. Note that the above statistical distance is exactly $\frac{1}{|R_q^\times|^m} \sum_{\boldsymbol{a} \in (R_q^\times)^m} \Delta_{\boldsymbol{a}}$, where $\Delta_{\boldsymbol{a}}$ is the distance to uniformity of $D_{\boldsymbol{a}}$. To prove the theorem, it therefore suffices to show a distance bound $\Delta_{\boldsymbol{a}} \leq 2\delta$, for all except a fraction $\leq 2^n(q-1)^{-\varepsilon n}$ of $\boldsymbol{a} \in (R_q^\times)^m$.

Now, the mapping $\boldsymbol{t} \mapsto \sum_i t_i a_i$ induces an isomorphism from the quotient group $\mathbb{Z}^{mn}/\boldsymbol{a}^\perp$ to its range (note that $\boldsymbol{a}^\perp$ is the kernel of $\boldsymbol{t} \mapsto \sum_i t_i a_i$). The latter is R_q, thanks to the invertibility of the a_i's. Therefore, the statistical distance $\Delta_{\boldsymbol{a}}$ is equal to the distance to uniformity of $\boldsymbol{t} \bmod \boldsymbol{a}^\perp$. By Lemma 4, we have $\Delta_{\boldsymbol{a}} \leq 2\delta$ if σ is greater than the smoothing parameter $\eta_\delta(\boldsymbol{a}^\perp)$ of $\boldsymbol{a}^\perp \subseteq \mathbb{Z}^{mn}$. To upper bound $\eta_\delta(\boldsymbol{a}^\perp)$, we apply Lemma 2, which reduces the task to lower

bounding the minimum of the dual lattice $\widehat{\boldsymbol{a}^{\perp}} = \frac{1}{q} \cdot L(\boldsymbol{a}^{\times})$, where $\boldsymbol{a}^{\times} \in (R_q^{\times})^m$ is in one-to-one correspondence with $\boldsymbol{a}$.

The following result is a direct consequence of Lemmata 2, 4, 7 and 8. Theorem 2 follows by taking $S = \emptyset$ and $\boldsymbol{c} = \boldsymbol{0}$.

Lemma 9. *Let $n \geq 8$ be a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q \geq 5$. Let $S \subseteq \{1, \ldots, n\}$, $m \geq 2$, $\varepsilon > 0$, $\delta \in (0, 1/2)$, $\boldsymbol{c} \in \mathbb{R}^{mn}$ and $\boldsymbol{t} \hookleftarrow D_{\mathbb{Z}^{mn}, \sigma, \boldsymbol{c}}$, with*

$$\sigma \geq \sqrt{n \ln(2mn(1 + 1/\delta))/\pi} \cdot q^{\frac{1}{m} + (m-1)\frac{|S|}{n} + \varepsilon}.$$

Then for all except a fraction $\leq 2^n (q-1)^{-\varepsilon n}$ of $\boldsymbol{a} \in (R_q^{\times})^m$, we have:

$$\Delta\Big[\boldsymbol{t} \bmod \boldsymbol{a}^{\perp}(I_S);\ U(R/\boldsymbol{a}^{\perp}(I_S))\Big] \leq 2\delta.$$

4 A revised key generation algorithm

We now use the results of the previous section on modular q-ary lattices to derive a key generation algorithm for `NTRUEncrypt`, where the generated public key follows a distribution for which Ideal-SVP reduces to R-LWE.

4.1 `NTRUEncrypt`'s key generation algorithm

The new key generation algorithm for `NTRUEncrypt` is given in Fig. 1. The secret key polynomials f and g are generated by using the Gentry *et al.* sampler of discrete Gaussians (see Lemma 5), and by rejecting so that the output polynomials are invertible modulo q. The Gentry *et al.* sampler may not exactly sample from discrete Gaussians, but since the statistical distance can be made exponentially small, the impact on our results is also exponentially small. Furthermore, it can be checked that our conditions on standard deviations are much stronger than the one in Lemma 5. From now on, we will assume we have a perfect discrete Gaussian sampler.

By choosing a large enough standard deviation σ, we can apply the results of the previous section and obtain the (quasi-)uniformity of the public key. We sample f of the form $p \cdot f' + 1$ so that it has inverse 1 modulo p, making the decryption process of `NTRUEncrypt` more efficient (as in the original `NTRUEncrypt` scheme). We remark that the rejection condition on f at Step 1 is equivalent to the condition $(f' \bmod q) \notin R_q^{\times} - p^{-1}$, where p^{-1} is the inverse of p in $R_q^{\times}$.

The following result ensures that for some appropriate choice of parameters, the key generation algorithm terminates in expected polynomial time.

Lemma 10. *Let $n \geq 8$ be a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q \geq 5$. Let $\sigma \geq \sqrt{n \ln(2n(1 + 1/\delta))/\pi} \cdot q^{1/n}$, for an arbitrary $\delta \in (0, 1/2)$. Let $a \in R$ and $p \in R_q^{\times}$. Then $\Pr_{f' \hookleftarrow D_{\mathbb{Z}^n, \sigma}}[(p \cdot f' + a \bmod q) \notin R_q^{\times}] \leq n(1/q + 2\delta)$.*

Inputs: $n, q \in \mathbb{Z}$, $p \in R_q^\times$, $\sigma \in \mathbb{R}$.
Output: A key pair $(sk, pk) \in R \times R_q^\times$.
1. Sample f' from $D_{\mathbb{Z}^n,\sigma}$; let $f = p \cdot f' + 1$; if $(f \bmod q) \notin R_q^\times$, resample.
2. Sample g from $D_{\mathbb{Z}^n,\sigma}$; if $(g \bmod q) \notin R_q^\times$, resample.
3. Return secret key $sk = f$ and public key $pk = h = pg/f \in R_q^\times$.

Fig. 1. Revised Key Generation Algorithm for `NTRUEncrypt`.

Proof. We are to bound the probability that $p \cdot f' + a$ belongs to $I := \langle q, \Phi_k \rangle$ by $1/q + 2\delta$, for any $k \leq n$. The result then follows from the Chinese Remainder Theorem and the union bound. We have $\mathcal{N}(I) = q$, so that $\lambda_1(I) \leq \sqrt{n} q^{1/n}$, by Minkowski's theorem. Since I is an ideal of R, we have $\lambda_n(I) = \lambda_1(I)$, and Lemma 1 gives that $\sigma \geq \eta_\delta(I)$. Lemma 4 then shows that $f \bmod I$ is within distance $\leq 2\delta$ to uniformity on R/I, so we have $p \cdot f' + a = 0 \bmod I$ (or, equivalently, $f' = -a/p \bmod I$) with probability $\leq 1/q + 2\delta$, as required. □

As a consequence of the above bound on the rejection probability, we have the following result, which ensures that the generated secret key is small.

Lemma 11. *Let $n \geq 8$ be a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q \geq 8n$. Let $\sigma \geq \sqrt{2n \ln(6n)/\pi} \cdot q^{1/n}$. The secret key polynomials f, g returned by the algorithm of Fig. 1 satisfy, with probability $\geq 1 - 2^{-n+3}$:*

$$\|f\| \leq 2n\|p\|\sigma \quad \textit{and} \quad \|g\| \leq \sqrt{n}\sigma.$$

If $\deg p \leq 1$, then $\|f\| \leq 4\sqrt{n}\|p\|\sigma$ with probability $\geq 1 - 2^{-n+3}$.

Proof. The probability under scope is lower than the probability of the same event without rejection, divided by the rejection probability. The result follows by combining Lemmata 3 and 10. □

4.2 Public key uniformity

In the algorithm of Fig. 1, the polynomials f' and g are independently sampled from the discrete Gaussian distribution $D_{\mathbb{Z}^n,\sigma}$ with $\sigma \geq \mathcal{P}oly(n) \cdot q^{1/2+\varepsilon}$ for an arbitrary $\varepsilon > 0$, but restricted (by rejection) to $R_q^\times - p^{-1}$ and $R_q^\times$, respectively. We denote by $D_{\sigma,z}^\times$ the discrete Gaussian $D_{\mathbb{Z}^n,\sigma}$ restricted to $R_q^\times + z$.

Here we apply the result of Section 3 to show that the statistical closeness to uniformity of a quotient of two distributions $(z + p \cdot D_{\sigma,y}^\times)$ for $z \in R_q$ and $y = -zp^{-1} \bmod q$. This includes the case of the public key $h = pg/f \bmod q$ computed by the algorithm of Fig. 1.

Theorem 3. *Let $n \geq 8$ be a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q \geq 5$. Let $\varepsilon > 0$ and $\sigma \geq 2n\sqrt{\ln(8nq)} \cdot q^{\frac{1}{2}+2\varepsilon}$. Let $p \in R_q^\times$, $y_i \in R_q$ and $z_i = -y_i p^{-1} \bmod q$ for $i \in \{1, 2\}$. Then*

$$\Delta\left[\frac{y_1 + p \cdot D_{\sigma,z_1}^\times}{y_2 + p \cdot D_{\sigma,z_2}^\times} \bmod q \ ; \ U\left(R_q^\times\right)\right] \ \leq \ 2^{3n} q^{-\lfloor \varepsilon n \rfloor}.$$

Proof. For $a \in R_q^\times$, we define $Pr_a = \Pr_{f_1,f_2}[(y_1+pf_1)/(y_2+pf_2) = a]$, where $f_i \hookleftarrow D^\times_{\sigma,z_i}$ for $i \in \{1,2\}$. We are to show that $|Pr_a - (q-1)^{-n}| \leq 2^{2n+5}q^{-\lfloor \varepsilon n \rfloor} \cdot (q-1)^{-n} =: \varepsilon'$ for all except a fraction $\leq 2^{2n}(q-1)^{-\varepsilon n}$ of $a \in R_q^\times$. This directly gives the claimed bound. The fraction of $a \in R_q^\times$ such that $|Pr_a - (q-1)^{-n}| \leq \varepsilon'$ is equal to the fraction of $\boldsymbol{a} = (a_1, a_2) \in (R_q^\times)^2$ such that $|Pr_{\boldsymbol{a}} - (q-1)^{-n}| \leq \varepsilon'$, where $Pr_{\boldsymbol{a}} = \Pr_{f_1,f_2}[a_1f_1 + a_2f_2 = a_1z_1 + a_2z_2]$. This is because $a_1f_1 + a_2f_2 = a_1z_1 + a_2z_2$ is equivalent to $(y_1+pf_1)/(y_2+pf_2) = -a_2/a_1$ (in $R_q^\times$), and $-a_2/a_1$ is uniformly random in $R_q^\times$ when $\boldsymbol{a} \hookleftarrow U((R_q^\times)^2)$.

We observe that $(f_1, f_2) = (z_1, z_2) =: \boldsymbol{z}$ satisfies $a_1f_1 + a_2f_2 = a_1z_1 + a_2z_2$, and hence the set of solutions $(f_1, f_2) \in R$ to the latter equation is $\boldsymbol{z} + \boldsymbol{a}^{\perp\times}$, where $\boldsymbol{a}^{\perp\times} = \boldsymbol{a}^\perp \cap (R_q^\times + q\mathbb{Z}^n)^2$. Therefore:

$$Pr_{\boldsymbol{a}} = \frac{D_{\mathbb{Z}^{2n},\sigma}(\boldsymbol{z} + \boldsymbol{a}^{\perp\times})}{D_{\mathbb{Z}^n,\sigma}(z_1 + R_q^\times + q\mathbb{Z}^n) \cdot D_{\mathbb{Z}^n,\sigma}(z_2 + R_q^\times + q\mathbb{Z}^n)}.$$

We now use the fact that for any $\boldsymbol{t} \in \boldsymbol{a}^\perp$ we have $t_2 = -t_1a_1/a_2$ so, since $-a_1/a_2 \in R_q^\times$, the ring elements t_1 and t_2 must belong to the *same* ideal I_S of R_q for some $S \subseteq \{1, \ldots, n\}$. It follows that $\boldsymbol{a}^{\perp\times} = \boldsymbol{a}^\perp \setminus \bigcup_{S\subseteq\{1,\ldots,n\},S\neq\emptyset} \boldsymbol{a}^\perp(I_S)$. Similarly, we have $R_q^\times + q\mathbb{Z}^n = \mathbb{Z}^n \setminus \bigcup_{S\subseteq\{1,\ldots,n\},S\neq\emptyset}(I_S + q\mathbb{Z}^n)$. Using the inclusion-exclusion principle, we obtain:

$$D_{\mathbb{Z}^{2n},\sigma}(\boldsymbol{z} + \boldsymbol{a}^{\perp\times}) = \sum_{S\subseteq\{1,\ldots,n\}} (-1)^{|S|} \cdot D_{\mathbb{Z}^{2n},\sigma}(\boldsymbol{z} + \boldsymbol{a}^\perp(I_S)), \quad (1)$$

$$\forall i \in \{1,2\} : D_{\mathbb{Z}^n,\sigma}(z_i + R_q^\times + q\mathbb{Z}^n) = \sum_{S\subseteq\{1,\ldots,n\}} (-1)^{|S|} \cdot D_{\mathbb{Z}^n,\sigma}(z_i + I_S + q\mathbb{Z}^n). \quad (2)$$

In the rest of the proof, we show that, except for a fraction $\leq 2^{2n}(q-1)^{-\varepsilon n}$ of $\boldsymbol{a} \in (R_q^\times)^2$:

$$D_{\mathbb{Z}^{2n},\sigma}(\boldsymbol{z} + \boldsymbol{a}^{\perp\times}) = (1+\delta_0) \cdot \frac{(q-1)^n}{q^{2n}},$$

$$\forall i \in \{1,2\} : \; D_{\mathbb{Z}^n,\sigma}(z_i + R_q^\times + q\mathbb{Z}^n) = (1+\delta_i) \cdot \frac{(q-1)^n}{q^n}.$$

where $|\delta_i| \leq 2^{2n+2}q^{-\lfloor \varepsilon n \rfloor}$ for $i \in \{0,1,2\}$. The bound on $|Pr_{\boldsymbol{a}} - (q-1)^{-n}|$ follows by a routine computation.

HANDLING (1). We note that, since $\boldsymbol{z} \in \mathbb{Z}^{2n}$, we have (for any $S \subseteq \{1, \ldots, n\}$):

$$D_{\mathbb{Z}^{2n},\sigma}(\boldsymbol{z}+\boldsymbol{a}^\perp(I_S)) = \frac{\rho_\sigma(\boldsymbol{z} + \boldsymbol{a}^\perp(I_S))}{\rho_\sigma(\mathbb{Z}^{2n})} = \frac{\rho_\sigma(\boldsymbol{z} + \boldsymbol{a}^\perp(I_S))}{\rho_\sigma(\boldsymbol{z} + \mathbb{Z}^{2n})} = D_{\mathbb{Z}^{2n},\sigma,-\boldsymbol{z}}(\boldsymbol{a}^\perp(I_S)).$$

For the terms of (1) with $|S| \leq \varepsilon n$, we apply Lemma 9 with $m = 2$. Since $|S|/n + \varepsilon \leq 2\varepsilon$, the Lemma 9 assumption on σ holds, with $\delta := q^{-n-\lfloor \varepsilon n \rfloor}$. We have $|R/\boldsymbol{a}^\perp(I_S)| = \det(\boldsymbol{a}^\perp(I_S)) = q^{n+|S|}$: Indeed, since $\boldsymbol{a} \in (R_q^\times)^2$, there are $q^{n-|S|}$ elements of $\boldsymbol{a}^\perp(I_S)$ in $[0, q-1]^{2n}$. We conclude that $|D_{\mathbb{Z}^{2n},\sigma,-\boldsymbol{z}}(\boldsymbol{a}^\perp(I_S)) -$

$q^{-n-|S|}| \le 2\delta$, for all except a fraction $\le 2^n(q-1)^{-\varepsilon n}$ of $\boldsymbol{a} \in (R_q^\times)^2$ (possibly corresponding to a distinct subset of $(R_q^\times)^2$ for each possible S).

For a term of (1) with $|S| > \varepsilon n$, we choose $S' \subseteq S$ with $|S'| = \lfloor \varepsilon n \rfloor$. Then we have $\boldsymbol{a}^\perp(I_S) \subseteq \boldsymbol{a}^\perp(I_{S'})$ and hence $D_{\mathbb{Z}^{2n},\sigma,-\boldsymbol{z}}(\boldsymbol{a}^\perp(I_S)) \le D_{\mathbb{Z}^{2n},\sigma,-\boldsymbol{z}}(\boldsymbol{a}^\perp(I_{S'}))$. By using with S' the above result for small $|S|$, we obtain $D_{\mathbb{Z}^{2n},\sigma,-\boldsymbol{z}}(\boldsymbol{a}^\perp(I_S)) \le 2\delta + q^{-n-\lfloor \varepsilon n \rfloor}$.

Overall, we have, except possibly for a fraction $\le 2^{2n}(q-1)^{-\varepsilon n}$ of $\boldsymbol{a} \in (R_q^\times)^2$:

$$\left| D_{\mathbb{Z}^{2n},\sigma}(\boldsymbol{z} + \boldsymbol{a}^{\perp\times}) - \sum_{k=0}^{n} (-1)^k \binom{n}{k} q^{-n-k} \right| \le 2^{n+1}\delta + 2 \sum_{k=\lceil \varepsilon n \rceil}^{n} \binom{n}{k} q^{-n-\lfloor \varepsilon n \rfloor}$$
$$\le 2^{n+1}(\delta + q^{-n-\lfloor \varepsilon n \rfloor}).$$

We conclude that $|\delta_0| \le \frac{q^{2n}}{(q-1)^n} 2^{n+1}(\delta + q^{-n-\lfloor \varepsilon n \rfloor}) \le 2^{2n+1}(\delta q^n + q^{-\lfloor \varepsilon n \rfloor})$, as required.

HANDLING (2). For the bounds on δ_1 and δ_2, we use a similar argument. Let $i \in \{1, 2\}$. The z_i term can be handled like like the $\boldsymbol{z}$ term of (1). We observe that for any $S \subseteq \{1, \ldots, n\}$, we have $\det(I_S + q\mathbb{Z}^n) = q^{|S|}$ and hence, by Minkowski's theorem, $\lambda_1(I_S + q\mathbb{Z}^n) \le \sqrt{n} \cdot q^{|S|/n}$. Moreover, since $I_S + q\mathbb{Z}^n$ is an ideal lattice, we have $\lambda_n(I_S + q\mathbb{Z}^n) = \lambda_1(I_S + q\mathbb{Z}^n) \le \sqrt{n} \cdot q^{|S|/n}$. Lemma 1 gives that $\sigma \ge \eta_\delta(I_S + q\mathbb{Z}^n)$ for any S such that $|S| \le n/2$, with $\delta := q^{-n/2}$. Therefore, by Lemma 4, for such an S, we have $|D_{\mathbb{Z}^n,\sigma,-z_i}(I_S + q\mathbb{Z}^n) - q^{-|S|}| \le 2\delta$.

For a term of (2) with $|S| > n/2$, we choose $S' \subseteq S$ with $|S'| = n/2$. By using with S' the above result for small $|S|$, we obtain $D_{\mathbb{Z}^n,\sigma,-z_i}(I_S + q\mathbb{Z}^n) \le D_{\mathbb{Z}^n,\sigma,-z_i}(I_{S'} + q\mathbb{Z}^n) \le 2\delta + q^{-n/2}$.

Overall, we have:

$$\left| D_{\mathbb{Z}^n,\sigma}(z_i + R_q^\times + q\mathbb{Z}^n) - \sum_{k=0}^{n} (-1)^k \binom{n}{k} q^{-k} \right| \le 2^{n+1}\delta + 2 \sum_{k=n/2}^{n} \binom{n}{k} q^{-n/2}$$
$$\le 2^{n+1}(\delta + q^{-n/2}),$$

which leads to the desired bound on δ_i (using $\varepsilon < 1/2$). This completes the proof of the theorem. □

5 `NTRUEncrypt` Revisited

Using our new results above, we describe a modification of `NTRUEncrypt` for which we can provide a security proof under a worst-case hardness assumption. We use $\Phi = x^n + 1$ with $n \ge 8$ a power of 2, $R = \mathbb{Z}[x]/\Phi$ and $R_q = R/qR$ with $q \ge 5$ prime such that $\Phi = \prod_{k=1}^{n} \Phi_k$ in R_q with distinct Φ_k's.

We define our modified `NTRUEncrypt` scheme with parameters n, q, p, α, σ as follows. The parameters n and q define the rings R and R_q. The parameter $p \in R_q^\times$ defines the plaintext message space as $\mathcal{P} = R/pR$. It must be a polynomial with 'small' coefficients with respect to q, but at the same time we require $\mathcal{N}(p) =$

$|\mathcal{P}| = 2^{\Omega(n)}$ so that many bits can be encoded at once. Typical choices as used in the original NTRUEncrypt scheme are $p = 3$ and $p = x + 2$, but in our case, since q is prime, we may also choose $p = 2$. By reducing modulo the px^i's, we can write any element of p as $\sum_{0 \le i < n} \varepsilon_i x^i p$, with $\varepsilon_i \in (-1/2, 1/2]$. Using the fact that $R = \mathbb{Z}[x]/(x^n + 1)$, we can thus assume that any element of $\mathcal{P}$ is an element of R with infinity norm $\le (\deg(p)+1) \cdot \|p\|$. The parameter α is the R-LWE noise distribution parameter. Finally, the parameter σ is the standard deviation of the discrete Gaussian distribution used in the key generation process (see Section 4).

- **Key generation.** Use the algorithm of Fig. 1 and return $sk = f \in R_q^\times$ with $f = 1 \bmod p$, and $pk = h = pg/f \in R_q^\times$.
- **Encryption.** Given message $M \in \mathcal{P}$, set $s, e \hookleftarrow \overline{\Upsilon}_\alpha$ and return ciphertext $C = hs + pe + M \in R_q$.
- **Decryption.** Given ciphertext C and secret key f, compute $C' = f \cdot C \in R_q$ and return $C' \bmod p$.

Fig. 2. The encryption scheme $\texttt{NTRUEncrypt}(n, q, p, \sigma, \alpha)$.

The correctness conditions for the scheme are summarized below.

Lemma 12. *If $\omega(n^{1.5} \log n)\alpha \deg(p)\|p\|^2\sigma < 1$ (resp. $\omega(n^{0.5} \log n)\alpha\|p\|^2\sigma < 1$ if $\deg p \le 1$) and $\alpha q \ge n^{0.5}$, then the decryption algorithm of* NTRUEncrypt *recovers M with probability $1 - n^{-\omega(1)}$ over the choice of s, e, f, g.*

Proof. In the decryption algorithm, we have $C' = p \cdot (gs + ef) + fM \bmod q$. Let $C'' = p \cdot (gs + ef) + fM$ computed in R (not modulo q). If $\|C''\|_\infty < q/2$ then we have $C' = C''$ in R and hence, since $f = 1 \bmod p$, $C' \bmod p = C'' \bmod p = M \bmod p$, i.e., the decryption algorithm succeeds. It thus suffices to give an upper bound on the probability that $\|C''\|_\infty > q/2$.

From Lemma 11, we know that with probability $\ge 1 - 2^{-n+3}$ both f and g have Euclidean norms $\le 2n\|p\|\sigma$ (resp. $4\sqrt{n}\|p\|\sigma$ if $\deg p \le 1$). This implies that $\|pf\|, \|pg\| \le 2n^{1.5}\|p\|^2\sigma$ (resp. $8\sqrt{n}\|p\|^2\sigma$), with probability $\ge 1 - 2^{-n+3}$. From Lemma 6, both pfe and pgs have infinity norms $\le 2\alpha q n^{1.5}\omega(\log n) \cdot \|p\|^2\sigma$ (resp. $8\alpha q\sqrt{n}\omega(\log n) \cdot \|p\|^2\sigma$), with probability $1 - n^{-\omega(1)}$. Independently:

$$\|fM\|_\infty \le \|fM\| \le \sqrt{n}\|f\|\|M\| \le 2 \cdot (\deg(p) + 1) \cdot n^2\|p\|^2\sigma \quad (\text{resp. } 8n\|p\|^2\sigma).$$

Since $\alpha q \ge \sqrt{n}$, we conclude that $\|C''\|_\infty \le (6 + 2\deg(p)) \cdot \alpha q n^{1.5}\omega(\log n) \cdot \|p\|^2\sigma$ (resp. $24\alpha q n^{0.5}\omega(\log n) \cdot \|p\|^2\sigma$), with probability $1 - n^{-\omega(1)}$. □

The security of the scheme follows by an elementary reduction from the decisional R-LWE$^\times_{\text{HNF}}$, exploiting the uniformity of the public key in $R_q^\times$ (Theorem 3), and the invertibility of p in R_q.

Lemma 13. *Suppose n is a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q = \omega(1)$. Let $\varepsilon, \delta > 0$, $p \in R_q^\times$ and $\sigma \ge 2n\sqrt{\ln(8nq)} \cdot q^{\frac{1}{2}+\varepsilon}$. If there exists an IND-CPA attack against* NTRUEncrypt *that runs in*

time T and has success probability $1/2+\delta$, then there exists an algorithm solving $\text{R-LWE}^{\times}_{\text{HNF}}$ *with parameters q and α that runs in time $T' = T + O(n)$ and has success probability $\delta' = \delta - q^{-\Omega(n)}$.*

Proof. Let $\mathcal{A}$ denote the given IND-CPA attack algorithm. We construct an algorithm $\mathcal{B}$ against $\text{R-LWE}^{\times}_{\text{HNF}}$ that runs as follows, given oracle $\mathcal{O}$ that samples from either $U(R_q^{\times} \times R_q)$ or $A^{\times}_{s,\psi}$ for some previously chosen $s \hookleftarrow \psi$ and $\psi \hookleftarrow \overline{\Upsilon}_{\alpha}$. Algorithm $\mathcal{B}$ first calls $\mathcal{O}$ to get a sample (h', C') from $R_q^{\times} \times R_q$. Then, algorithm $\mathcal{B}$ runs $\mathcal{A}$ with public key $h = p \cdot h' \in R_q$. When $\mathcal{A}$ outputs challenge messages $M_0, M_1 \in \mathcal{P}$, algorithm $\mathcal{B}$ picks $b \hookleftarrow U(\{0,1\})$, computes the challenge ciphertext $C = p \cdot C' + M_b \in R_q$, and returns C to $\mathcal{A}$. Eventually, when $\mathcal{A}$ outputs its guess b' for b, algorithm $\mathcal{B}$ outputs 1 if $b' = b$ and 0 otherwise.

The h' used by $\mathcal{B}$ is uniformly random in $R_q^{\times}$, and therefore so is the public key h given to $\mathcal{A}$, thanks to the invertibility of p modulo q. Thus, by Theorem 3, the public key given to $\mathcal{A}$ is within statistical distance $q^{-\Omega(n)}$ of the public key distribution in the genuine attack. Moreover, since $C' = h \cdot s + e$ with $s, e \hookleftarrow \psi$, the ciphertext C given to $\mathcal{A}$ has the right distribution as in the IND-CPA attack. Overall, if $\mathcal{O}$ outputs samples from $A^{\times}_{s,\psi}$, then $\mathcal{A}$ succeeds and $\mathcal{B}$ returns 1 with probability $\geq 1/2 + \delta - q^{-\Omega(n)}$.

Now, if $\mathcal{O}$ outputs samples from $U(R_q^{\times} \times R_q)$, then, since $p \in R_q^{\times}$, the value of $p \cdot C'$ and hence C, is uniformly random in R_q and independent of b. It follows that $\mathcal{B}$ outputs 1 with probability $1/2$. The claimed advantage of $\mathcal{B}$ follows. □

By combining Lemmata 12 and 13 with Theorem 1 we obtain our main result.

Theorem 4. *Suppose n is a power of 2 such that $\Phi = x^n + 1$ splits into n linear factors modulo prime $q = \mathcal{P}oly(n)$ such that $q^{\frac{1}{2}-\varepsilon} = \omega(n^{3.5} \log^2 n \deg(p) \|p\|^2)$ (resp. $q^{\frac{1}{2}-\varepsilon} = \omega(n^4 \log^{1.5} n \deg(p) \|p\|^2)$), for arbitrary $\varepsilon \in (0, 1/2)$ and $p \in R_q^{\times}$. Let $\sigma = 2n\sqrt{\ln(8nq)} \cdot q^{\frac{1}{2}+\varepsilon}$ and $\alpha^{-1} = \omega(n^{1.5} \log n \deg(p) \|p\|^2 \sigma)$. If there exists an IND-CPA attack against* $\texttt{NTRUEncrypt}(n, q, p, \sigma, \alpha)$ *which runs in time $T = \mathcal{P}oly(n)$ and has success probability $1/2 + 1/\mathcal{P}oly(n)$ (resp. time $T = 2^{o(n)}$ and success probability $1/2 + 2^{-o(n)}$), then there exists a $\mathcal{P}oly(n)$-time (resp. $2^{o(n)}$-time) quantum algorithm for γ-*Ideal-SVP *with $\gamma = O(n^4 \log^{2.5} n \deg(p) \|p\|^2 q^{\frac{1}{2}+\varepsilon})$ (resp. $\gamma = O(n^5 \log^{1.5} n \deg(p) \|p\|^2 q^{\frac{1}{2}+\varepsilon})$). Moreover, the decryption algorithm succeeds with probability $1 - n^{-\omega(1)}$ over the choice of the encryption randomness.*

In the case where $\deg p \leq 1$, the conditions on q for polynomial-time (resp. subexponential) attacks in Theorem 4 may be relaxed to $q^{\frac{1}{2}-\varepsilon} = \omega(n^{2.5} \log^2 n \cdot \|p\|^2)$ (resp. $q^{\frac{1}{2}-\varepsilon} = \omega(n^3 \log^{1.5} n \cdot \|p\|^2)$) and the resulting Ideal-SVP approximation factor may be improved to $\gamma = O(n^3 \log^{2.5} n \cdot \|p\|^2 q^{\frac{1}{2}+\varepsilon})$ (resp. $\gamma = O(n^4 \log^{1.5} n \cdot \|p\|^2 q^{\frac{1}{2}+\varepsilon})$). Overall, by choosing $\varepsilon = o(1)$, the smallest q for which the analysis holds is $\widetilde{\Omega}(n^5)$ (resp. $\widetilde{\Omega}(n^6)$), and the smallest γ that can be obtained is $\widetilde{O}(n^{5.5})$ (resp. $\widetilde{O}(n^7)$).

Acknowledgements. We thank G. Hanrot, V. Lyubashevsky, Khoa T. T. Nguyen, C. Peikert, O. Regev, I. Shparlinski, J. Silverman and F. Vercauteren

for helpful discussions. The authors were partly supported by the LaRedA ANR grant, an Australian Research Fellowship (ARF) from the Australian Research Council under Discovery Grant DP0987734, a Macquarie University Research Fellowship, and ARC Discovery Grant DP110100628.

References

1. S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *Proc. of Eurocrypt*, volume 6110 of *LNCS*, pages 553–572. Springer, 2010.
2. M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the 28th Symposium on the Theory of Computing (STOC 1996)*, pages 99–108. ACM, 1996.
3. B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Proc. of CRYPTO*, volume 5677 of *LNCS*, pages 595–618. Springer, 2009.
4. W. D. Banks and I. E. Shparlinski. Distribution of inverses in polynomial rings. *Indagationes Mathematicae*, 12(3):303–315, 2001.
5. D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *Proc. of Eurocrypt*, volume 6110 of *LNCS*, pages 523–552. Springer, 2010.
6. D. Coppersmith and A. Shamir. Lattice attacks on NTRU. In *Proc. of Eurocrypt*, volume 1233 of *LNCS*, pages 52–61. Springer, 1997.
7. J. von zur Gathen and J. Gerhardt. *Modern Computer Algebra, 2nd edition*. Cambridge University Press, 2003.
8. C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. Manuscript available at `http://crypto.stanford.edu/craig`.
9. C. Gentry. Fully homomorphic encryption using ideal lattices. In *Proc. of STOC*, pages 169–178. ACM, 2009.
10. C. Gentry. Toward basing fully homomorphic encryption on worst-case hardness. In *Proc. of CRYPTO*, volume 6223 of *LNCS*, pages 116–137. Springer, 2010.
11. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proc. of STOC*, pages 197–206. ACM, 2008.
12. N. Higham. *Accuracy and Stability of Numerical Algorithms, 2nd edition*. SIAM, 2002.
13. J. Hoffstein, N. Howgrave-Graham, J. Pipher, and W. Whyte. Practical lattice-based cryptography: `NTRUEncrypt` and `NTRUSign`, 2009. Chapter of [27].
14. J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: a new high speed public key cryptosystem. Preprint; presented at the rump session of Crypto'96, 1996.
15. J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: a ring based public key cryptosystem. In *Proc. of ANTS*, volume 1423 of *LNCS*, pages 267–288. Springer, 1998.
16. J. Hoffstein, J. Pipher, and J. H. Silverman. NSS: An NTRU lattice-based signature scheme. In *Proc. of Eurocrypt*, volume 2045 of *LNCS*. Springer, 2001.
17. N. A. Howgrave-Graham, J. H. Silverman, and W. Whyte. Choosing parameter sets for NTRUEncrypt with NAEP and SVES-3. In *Proc. of CT-RSA*, volume 3376 of *LNCS*. Springer, 2005.
18. IEEE P1363. Standard specifications for public-key cryptography. `http://grouper.ieee.org/groups/1363/`.

19. V. Lyubashevsky and D. Micciancio. Generalized compact knapsacks are collision resistant. In *Proc. ICALP*, volume 4052 of *LNCS*, pages 144–155. Springer, 2006.
20. V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. SWIFFT: A modest proposal for FFT hashing. In *Proc. of FSE*, volume 5086 of *LNCS*, pages 54–72. Springer, 2008.
21. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In *Proc. of Eurocrypt*, volume 6110 of *LNCS*, pages 1–23. Springer, 2010.
22. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings, 2011. Draft for the extended version of [21], dated 01/02/2011.
23. D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Comput. Complexity*, 16(4):365–411, 2007.
24. D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput*, 37(1):267–302, 2007.
25. D. Micciancio and O. Regev. Lattice-based cryptography. In *Post-Quantum Cryptography, D. J. Bernstein, J. Buchmann, E. Dahmen (Eds)*, pages 147–191. Springer, 2009.
26. D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In *Proc. of STOC*, pages 351–358. ACM, 2010.
27. P. Q. Nguyen and B. Vallée (editors). *The LLL Algorithm: Survey and Applications*. Information Security and Cryptography. Springer, 2009.
28. C. Peikert. Limits on the hardness of lattice problems in ℓ_p norms. *Comput. Complexity*, 2(17):300–351, 2008.
29. C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proc. of STOC*, pages 333–342. ACM, 2009.
30. C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *Proc. of TCC*, volume 3876 of *LNCS*, pages 145–166. Springer, 2006.
31. R. A. Perlner and D. A. Cooper. Quantum resistant public key cryptography: a survey. In *Proc. of IDtrust*, pages 85–93. ACM, 2009.
32. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.
33. O. Regev. The learning with errors problem, 2010. Invited survey in CCC 2010, available at `http://www.cs.tau.ac.il/~odedr/`.
34. C. P. Schnorr. A hierarchy of polynomial lattice basis reduction algorithms. *Theor. Comput. Science*, 53:201–224, 1987.
35. D. Stehlé, R. Steinfeld, K. Tanaka, and K. Xagawa. Efficient public key encryption based on ideal lattices. In *Proc. of Asiacrypt*, volume 5912 of *LNCS*, pages 617–635. Springer, 2009.
36. T. Xylouris. On Linnik's constant, 2009. Available at `http://arxiv.org/abs/0906.2749` (in German).