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
Oded REGEV (absent), Directeur de recherche, CNRS
Bruno SALVY, Directeur de recherche, INRIA
Arne STORJOHANN, Associate professor, University of Waterloo
Brigitte VALLÉE, Directrice de recherche, CNRS
Chee K. YAP (absent), 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

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;

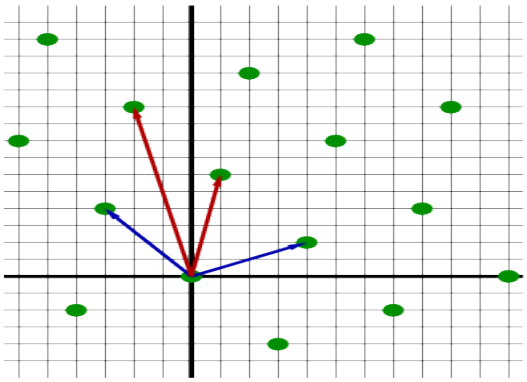

Figure 1.1: A two-dimensional lattice along with two of its bases.

- 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:

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

As $\lambda_1(L) \leq \lambda_2(L) \leq \ldots, \leq \lambda_n(L)$, 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.

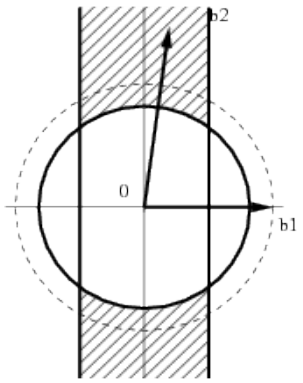

Figure 1.2: The hashed area is the set of possible locations for $(\mathbf{b}_1, \mathbf{b}_2)$ to be δ-LLL-reduced.

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 Vaikuntanathan [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

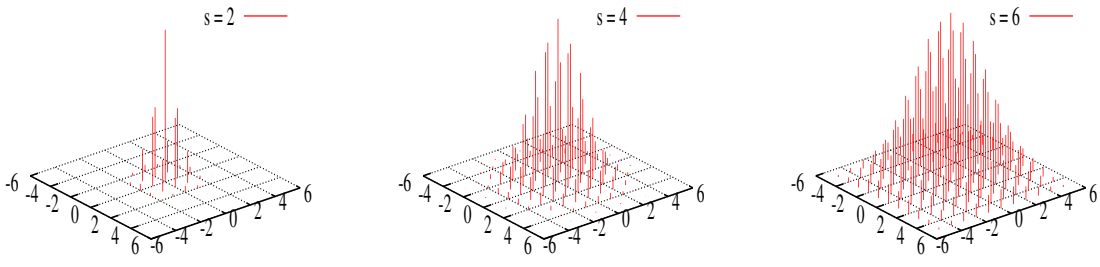

Figure 1.3: Three discrete Gaussian distributions with support $\mathbb{Z}^2$ and centre $\mathbf{0}$, but different standard deviations s.

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.

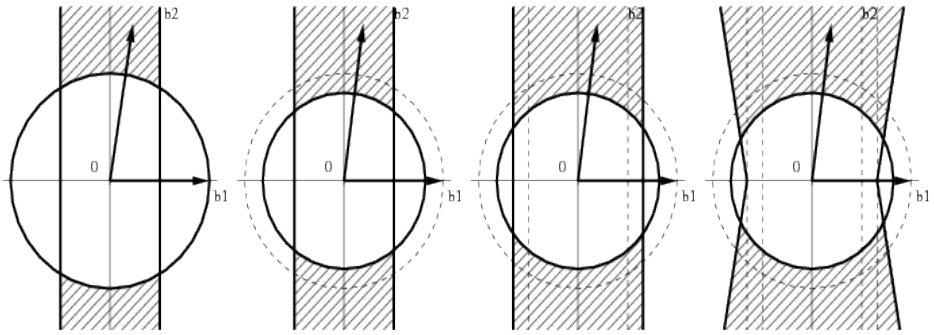

Figure 2.3: The hashed area is the set of vectors $\mathbf{b}_2$ such that $(\mathbf{b}_1, \mathbf{b}_2)$ is (from left to right) $(1, 0, 0)$-LLL, $(\delta, 0, 0)$-LLL, $(\delta, \eta, 0)$-LLL and (δ, η, θ)-LLL.

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 $\diamond(\overline{\delta} \cdot \diamond(\overline{r}_{\kappa-1\kappa-1}^2)) \leq s$, 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 $(\mathbf{b}_i)_{i\leq n}$ 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 $i \leq n$, 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 $\sum_{1\leq j\leq n} \ell_j \leq A^2$, $S := S \cup \{\mathbf{x}\}$, $x_1 := x_1 + 1$.
7. $\quad$ If $i \neq 1$ and $\sum_{j\geq i} \ell_j \leq A^2$, $i := i - 1$, $x_i := \left\lceil t_i - \sum_{j>i}(x_j \mu_{ji}) - \sqrt{\frac{A^2 - \sum_{j>i} \ell_j}{\|\mathbf{b}_i^*\|^2}} \right\rceil$.
8. $\quad$ If $\sum_{j\geq i} \ell_j > A$, 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 $i \leq n$):

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

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.

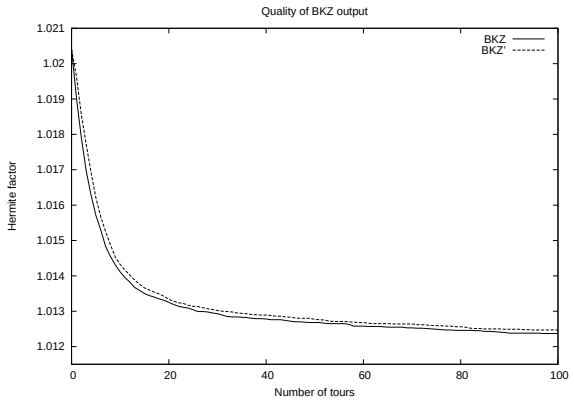

Figure 3.2: Evolution of the Hermite factor $\frac{\|\mathbf{b}_1\|}{(\det L)^{1/n}}$ during the execution of BKZ and BKZ'.

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 $\widetilde{\mathrm{L}}^1$ 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 $\widetilde{\mathrm{L}}^1$, 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 $\widetilde{\mathrm{L}}^1$, 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 $\tilde{O}(t)$, 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 rot$_f(\mathbf{g})$. The matrix rot$_f(\mathbf{g})$ is defined as follows: If $r \in \mathbb{Z}[x]/f$, then 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 $A = \text{rot}_f(\mathbf{a})$ 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 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, 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 Υ_α. The important facts to be remembered are that sampling from Υ_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 $\psi \hookleftarrow \overline{\Upsilon}_\alpha$ 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 $1/\mathrm{poly}(n)$ (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 $s, e \hookleftarrow \phi_\alpha \hookleftarrow \Upsilon_\alpha$ 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.

Résumé

Les réseaux Euclidiens sont un riche objet algébrique qui apparaît dans des contextes variés en mathématiques et en informatique. Cette thèse considère plusieurs aspects algorithmiques des réseaux. Le concept de réduction d'une base d'un réseau est étudié minutieusement : nous couvrons en particulier le spectre complet des compromis qualité-temps des algorithmes de réduction. D'une part, nous présentons et analysons des algorithmes rapides pour trouver une base assez courte (base LLL-réduite) d'un réseau donné arbitraire. D'autre part, nous proposons de nouvelles analyses pour des algorithmes (plus lents) permettant de calculer des bases très courtes (bases HKZ et BKZ-réduites). Cette étude des algorithmes de résolution efficace de problèmes portant sur les réseaux est complétée par une application constructive exploitant leur difficulté apparente. Nous proposons et analysons des schémas cryptographiques, dont la fonction de chiffrement NTRU, et les prouvons au moins aussi difficiles à casser que de résoudre des problèmes pires-cas bien spécifiés portant sur les réseaux.

Mots-clés. Réseaux Euclidiens, réductions LLL/HKZ/BKZ, cryptographie reposant sur les réseaux Euclidiens, algorithmes hybrides numériques-algébriques, analyse d'algorithmes.

Abstract

Euclidean lattices are a rich algebraic object that occurs in a wide variety of contexts in mathematics and in computer science. The present thesis considers several algorithmic aspects of lattices. The concept of lattice basis reduction is thoroughly investigated: in particular, we cover the full range of time-quality trade-offs of reduction algorithms. On the first hand, we describe and analyse fast algorithms for finding a relatively short basis (LLL-reduced basis) of an arbitrary given lattice. On the second hand, we propose novel analyses for (slower) algorithms that compute very short bases (HKZ-reduced and BKZ-reduced bases). This study on how to efficiently solve algorithmic problems on lattices is completed by a constructive application exploiting their apparent hardness. We propose and analyze cryptographic schemes, including the NTRU encryption function, and prove them at least as secure as well-specified worst-case problems on lattices.

Keywords. Euclidean lattices, LLL/HKZ/BKZ reductions, lattice-based cryptography, computer algebra, hybrid symbolic-numeric algorithms, analysis of algorithms.