CERTIFIED LATTICE-REDUCTION

THOMAS ESPITAU AND ANTOINE JOUX

Abstract. Quadratic form reduction and lattice reduction are fundamental tools in computational number theory and in computer science, especially in cryptography. The celebrated Lenstra–Lenstra–Lovász reduction algorithm (so-called LLL) has been improved in many ways through the past decades and remains one of the central method used for reducing *integral* lattice basis. In particular, its floating-point variants—where the rational arithmetic required by Gram–Schmidt orthogonalization is replaced by floating-point arithmetic—are now the fastest known. Nonetheless, the systematic study of the reduction theory of *real* quadratic forms or more generally of real lattices—in particular on the precision needed to represent lattices in such a way the reduction can soundly operate—is not widely reprented in the litteratre despite its intrinsic interest. In this work, we present a sound adaptive-precision version of a generalized LLL algorithm working for lattices endowed with an arbitrary scalar product, as well as a theoretical analysis of the representation needed to perform such a reduction. In this framework, floating-point arithmetic is replaced by Interval Arithmetic. The inherent certification property of Interval Arithmetic enables runtime detection of precision defects in numerical computations and accordingly makes it possible to run the reduction algorithms with guaranteed nearly optimal precision. We then present some applications of these results to algebraic number theory, and more precisely develop certified techniques for an algorithmic insight of the Geometry of Numbers in number fields.

1. Introduction

In a general setting, a *lattice* Λ is a free $\mathbf{Z}$-module of finite rank, endowed with a positive-definite quadratic form on its ambient space $\Lambda \otimes_{\mathbf{Z}} \mathbf{Q}$, as presented for instance in [LS17]. This formalism encompasses the well-known *Euclidean lattices* when taking the canonical scalar product of $\mathbf{Q}^d$, but also lattices arising from ideals in rings of integers, projective modules over orders in number fields. The rank of the lattice is defined intrinsically as the dimension of the vector space $\Lambda \otimes_{\mathbf{Z}} \mathbf{Q}$. Equivalently, by definition of a finitely-generated free module, there exists a finite set of vectors $b_1, \ldots, b_{\operatorname{rk}\Lambda} \in \Lambda$ such that $\Lambda = \bigoplus_{i=1}^{\operatorname{rk}\Lambda} b_i\mathbf{Z}$. Such a family is called a basis of the lattice and is not unique. In fact, as soon as $\operatorname{rk}\Lambda \geq 2$ there are infinitely many bases of Λ. Among those some have interesting properties, such as having reasonably small vectors and low orthogonality defect. They are called *reduced bases* and finding them is the goal of *lattice reduction* theory. This has been crucial in several fields of computer science and mathematics, for instance in cryptology, where lattices have been used to break many public-key cryptosystems in the last decades:

Key words and phrases. Lattice Reduction, Quadratic forms reduction, Algorithmic number theory, Numerical analysis.

This work has been supported in part by the European Union as H2020 Programme under grant agreement number ERC-669891.

knapsack cryptosystems [LO85] or RSA in specific settings thanks to Coppersmith's method [B^+99]. Moreover, numerous algorithms arising in algebraic number theory rely heavily on lattice reduction, such as the computation of normal forms of integral matrices (see [Jäg05] for the Hermite Normal Form and [HMM98] for the Smith Normal Form), class group computations in a number field [GJ16, BF13], resolution of the so-called principal ideal problem [EFGK16] or even the enumeration of points of small height near algebraic curves [Elk00]. A revolutionary application was given in 1985 by Odlyzko and Ed Riele in [OR85] to disproof the Mertens' conjecture on the cumulative function of the Möbius function. The link between lattices and quadratic forms appears quite clearly when considering the *Gram-matrix* of a basis $\mathcal{B} = \{b_1, \ldots, b_d\}$, that is the real symmetric matrix $\mathcal{G} = (\langle b_i, b_j \rangle)_{i,j}$: the $\mathrm{Sl}_d(\mathbf{Z})$-action by right-multiplication on $\mathcal{B}$ directly translates to the usual $\mathrm{Sl}_d(\mathbf{Z})$-action on the coefficients of the n-ary quadratic form represented by the matrix $\mathcal{G}$. Finding $\mathrm{Sl}_d(\mathbf{Z})$-equivalent quadratic forms with small coefficients is the target of quadratic form reduction. Hence any quadratic form reduction algorithm turns out to be also a lattice reduction algorithm, by applying the unimodular transformations used on the form directly to the basis.

The study of these reduction problems is not recent and goes back to the early works of Lagrange and Gauss, for integral binary quadratic forms[1], and the introduction of the so-called Gauss algorithm. This procedure can be seen as a 2-dimensional extension of the Euclid algorithm for computing the greatest common divisor of two integers. In 1850, Hermite published the first reduction algorithm for arbitrary dimension[2]. A century later, in 1982, Lenstra, Lenstra and Lovász designed the *LLL algorithm* [LLL82], with the polynomial factorization problem as an application, after the celebrated work of Lenstra on integer programming [Len83]. This algorithm is a milestone in the history of lattice reduction algorithms, being the first algorithm whose running-time is polynomial in terms of the dimension. This work has been improved in multiple ways by Kaltofen [Kal83], Schnorr [Sch87], and Gama and Nguyen [GN08] among others, decreasing the time complexity or improving the quality of the reduction.

Interval arithmetic is a representation of reals by intervals—whose endpoints are floating-point numbers—that contain them. Arithmetic operations, in particular the basic operations $+, -, \times, \div$ can be redefined in this context. The main interest of this representation lies in its *certification* property: if real numbers are represented by intervals, the interval resulting from the evaluation of an algebraic expression contains the exact value of the evaluated expression.

If the birth of Lattice Reduction is well-dated, it is not the case of Interval Arithmetic. For some authors, it has been introduced by R. Moore in 1962 in his Ph.D. thesis [Moo62]. For others, it can be dated back to 1958 in an article of T. Sunaga [Sun09] which describes an algebraic interpretation of the lattice of real intervals, or even sooner in 1931 as a proposal in the Ph.D. thesis [You31]

[1]Hence their works corresponds to the reduction of dimension two lattices with integral coefficients

[2]*Stricto sensu*, he presented two algorithms, one for proving the existence of the so-called Hermite constant which bounds the length of the shortest vector of a lattice, and the other which allows finding bases with low orthogonality defect.

of R.C. Young at Cambridge. Nonetheless, its development and industrial applications had to wait until 1980, and the momentum of U. Kulisch in Karlsruhe, leading IBM to develop a specific instruction set and a compiler natively integrating Interval Arithmetic. Its main asset—calculating directly on sets—is nowadays used to deterministically determine the global extrema of a continuous function [RR88] or localizing the zeroes of a function and (dis)proving their existence [JKDW01]. Another application of Interval Arithmetic is to be able to detect lack of precision at run-time of numerical algorithms, thanks to the guarantees it provides on computations.

This last application can lead to the design of adaptive precision numerical algorithms. In the present paper, we propose to transform and generalize the LLL algorithm into an adaptive precision version, which reduces arbitrary lattices[3]. This work leads to design a *certified* reduction algorithm whose main assets are:

- A general framework to algorithmically represents arbitrary real lattices through Interval Arithmetic, together with a theoretical analysis on the precision required to handle these representations – especially those manipulated by algorithms appearing in algebraic number theory.
- An algorithmic certification of the reduceness of bases of arbitrary real lattices
- A certified version of the LLL algorithm: its execution flow is the same as the original integral version of LLL. Thus, it can be used to perform experiments on the behavior of LLL and on the properties of the output basis.

All in all this framework leads to a sound and provable algorithmic version of the Minkovsky theory for the reduction of *real* quadratic forms.

Organisation of the paper. In Section 2 we briefly introduce reduction theory and present the L^2 variant of the LLL algorithm. Section 3 aims at describing the basics of Interval Arithmetic used in Section 4 to handle the problem of representation of real lattices. The framework of this latter section is then used in Section 5 to derive a certified reduction algorithm for real lattices. Section 6 presents applications of this methodology to algorithmic number theory.

Notations and conventions.

General notations. The bold capitals $\mathbf{Z}$, $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{C}$ refer as usual to the ring of integers and respectively the field of rational, real and complex numbers. Given a real number x, the integral roundings *floor*, *ceil* and *round to nearest integer* are denoted respectively by $\lfloor x \rfloor$, $\lceil x \rceil$, $\lfloor x \rceil$. Note that the rounding operator is inherently ambiguous when operating on half-integers, and as such a convention has to be chosen[4]. These operators are extended straightforwardly to operate on vectors and matrices by point-wise composition. The complex conjugation of $z \in \mathbf{C}$ is denoted by the usual bar $\bar{z}$ whereas the real and imaginary parts of a complex z are indicated by respectively $\Re(z)$ and $\Im(z)$. All logarithms are taken in base 2.

[3]The original LLL algorithm only reduces integral Euclidean lattices, that is sublattices of $\mathbf{Z}^d$.

[4]Two rounding rules usually used are on the one hand to round half-integers to the nearest even integer, or on the other hand to round them to the closest lower integer.

Matrix and norms. For a field $\mathbf{K}$, let us denote by $\mathbf{K}^{d\times d}$ the space of square matrices of size d over $\mathbf{K}$, $\mathrm{Gl}_d(\mathbf{K})$ its group of invertibles and $\mathcal{S}_d(\mathbf{K})$ its subspace of symmetric matrices. For a complex matrix A, we write $A^\dagger$ for its conjugate-transpose. For a vector v, we denote by $\|v\|_\infty$ its absolute (or infinity) norm, that is the maximum of the absolute value of its coefficients. We similarly define the matrix *max-norm* $\|B\|_{\max} = \max_{1\le i\le j\le d} |B_{i,j}|$, for any matrix B.

Computational setting. The generic complexity model used in this work is the random-access machine (RAM) model and the computational cost is measured in bits operations. $M(k)$ denotes the complexity of the multiplication of two integers of bit-length at most k.

2. A bird's eye view on Reduction Theory

2.1. **Remarks on the orthogonalization of vectors.**

Let us fix an Euclidean space $(E, \langle\cdot,\cdot\rangle)$, that is, a real vector E space endowed with a positive-definite quadratic form $\langle\cdot,\cdot\rangle : E\times E \to \mathbf{R}$. Recall that two vectors $x, y \in E$ are said to be orthogonal—with regards to the form $\langle\cdot,\cdot\rangle$—if $\langle x, y\rangle = 0$. More generally a family of vectors is orthogonal if its elements are pairwise orthogonal.

Now consider $S = (v_1, \dots, v_d)$ a basis of E. The *flag* $\mathcal{F}_S$ associated to S is the data of the finite increasing chain of subspaces:

$$\mathbf{R}v_1 \subset \mathbf{R}v_1 \oplus \mathbf{R}v_2 \subset \cdots \subset \bigoplus_{i=1}^{r} v_i\mathbf{R}.$$

The orthogonal complement $S^\perp$ is defined as the space $\{x\in E \mid \forall i,\ \langle x, v_i\rangle = 0\}$. Denote by π_i the orthogonal projection on $(b_1, \dots, b_{i-1})^\perp$, with the convention that $\pi_1 = \mathrm{Id}$. The Gram–Schmidt process—shorthanded in GSO—is an algorithmic method for orthogonalizing S while preserving its flag, that is constructing the orthogonal set $S^* = (\pi_1(v_1), \dots, \pi_r(v_r))$. The computation S^* can done inductively as follow:

$$\pi_1(v_1) = v_1$$
$$\forall 1 < i \le r, \quad \pi_i(v_i) = v_i - \sum_{j=1}^{i-1} \frac{\langle v_i, \pi_j(v_j)\rangle}{\langle \pi_j(v_j), \pi_j(v_j)\rangle}.$$

Define the *Gram matrix* associated to a family of vector $S = (v_1, \dots, v_r)$ as the matrix of scalar products: $\mathcal{G}_S = (\langle v_i, v_j\rangle)_{1\le i,j\le n}$. The (co)volume of S is defined as norm of the exterior product vector $v_1\wedge\cdots\wedge v_n$, that is as the square root of the Gram determinant $\det \mathcal{G}_S$. It can also be easily computed with the Gram-Schmidt vectors S^* as:

$$\mathrm{covol}(S) = \prod_{i=1}^{r} \|\pi_i(v_i)\|$$

2.2. **Lattices and reduction.**

Definition 2.1. *A (real)* lattice Λ *is a finitely generated free* $\mathbf{Z}$*-module, endowed with a positive-definite quadratic form* $\langle\cdot,\cdot\rangle$ *on its ambient space* $\Lambda\otimes_{\mathbf{Z}}\mathbf{Q}$*, making* Λ *discrete for the induced norm. By definition of a finitely-generated free module, there exists a finite family* $b_1, \dots, b_d \in \Lambda^d$ *such that* $\Lambda = \bigoplus_{i=1}^d b_i\mathbf{Z}$*, called a* basis *of* Λ*.*

We may omit to write down the quadratic form to refer to a lattice Λ when any ambiguity is removed by the context. In all of this section, $\|.\|$ stands for the Euclidean norm induced by $\langle\cdot,\cdot\rangle$, unless stated otherwise.

Two different bases of the same lattice Λ are related by a unimodular transformation, which is a linear transformation represented by an element of $\mathrm{Gl}_d(\mathbf{Z})$, set of $d \times d$ integer-valued matrices of determinant ± 1. The space of (real) lattices is then homeomorphic to the quotient ${\mathrm{Gl}_d(\mathbf{R})}\big/{\mathrm{Gl}_d(\mathbf{Z})}$. Thus, algorithms acting on lattice bases can be seen as sequences of unimodular transformations to achieve their goal. Among these procedures, so-called reduction algorithms are of the utmost importance.

Reduction theory aims at giving general techniques to construct preferred and congenial classes of bases, proving that for any lattice, one can efficiently find *quasi-orthogonal* bases with controlled norm vectors. Roughly speaking, a basis $(b_1, \ldots, b_d)$ is considered to be *reduced* with regards to a certain parameter $\delta < 1$ if:

- $\|\pi_{i+1}(b_{i+1})\| \leq \delta \|\pi_i(b_i)\|$
- When decomposing b_i in $\pi_i(b_i) + \sum_{j<i} \alpha_{i,j}\pi_j(b_j)$ we get $|\alpha_{i,j}| \leq \frac{1}{2}$,

where π_i is the orthogonal projection on $(b_{i+1}, \ldots, b_d)^\perp$ in the space $(\Lambda \otimes_{\mathbf{Z}} \mathbf{Q}, \langle\cdot,\cdot\rangle)$, as introduced in Section 2.1. The reduction notion introduced here actually corresponds to the Siegel-reduction, and not LLL-reduction, but its introduction eases the presentation of the notion.

In order to prove that such bases exist, let outline a constructive proof which lays the foundation of the LLL-reduction algorithm. Starting with the data of the initial basis $(b_1, \ldots, b_d)$, we proceed iteratively. Each step starts by considering the set of projections $(\pi_1(b_1), \ldots, \pi_d(b_d))$. Thanks to the geometry of the ring $\mathbf{Z}$ and by cleverly choosing a linear combinations of the $b_{i+1}, \ldots, b_d$, we can suppose that the following decomposition holds:

$$b_i = \pi_i(b_i) + \sum_{j<i} \alpha_{i,j}\pi_j(b_j) \qquad \text{with } |\alpha_{i,j}| \leq \frac{1}{2}.$$

To enforce the decrease condition on the norms of the $\pi_1(b_1), \ldots, \pi_d(b_d)$, we greedily swap b_i and b_{i+1} as soon as the ratio $\|\pi_i(b_i)\|/\|\pi_{i+1}(b_{i+1})\|$ gets greater than the desired constant δ.

2.3. The LLL reduction algorithm.

The above-introduced procedure is in substance the basic ingredient of the reduction presented in 1982 by Lenstra, Lenstra and Lovász [LLL82]. They proposed a notion called *LLL-reduction* conjointly with a polynomial time algorithm. Their reduction notion is formally defined as follows:

Definition 2.2 (LLL reduction)**.** *A basis $\mathcal{B}$ of a lattice is said to be δ-**LLL-reduced** for certain parameters $1/4 < \delta \leq 1$, if the following conditions are satisfied:*

$$(1) \qquad \forall i < j, \quad |\langle b_j, \pi_i(b_i)\rangle| \leq \frac{1}{2}\|\pi_i(b_i)\|^2 \quad \textit{(Size-Reduction condition)}$$

$$(2) \qquad \forall i, \quad \delta\|\pi_i(b_i)\|^2 \leq \left(\|\pi_{i+1}(b_{i+1})\|^2 + \frac{\langle b_{i+1}, \pi_i(b_i)\rangle}{\|\pi_i(b_i)\|^2}\right) \quad \textit{(Lovász condition)}$$

Remarking that the orthogonal projection $\pi_i(b_i)$ can be computed iteratively by the Gram-Schmidt orthogonalization process, we can translate directly the sketch of proof into an algorithm, presented in Figure 1.

Remark. *Note that the original algorithm from [LLL82] only deals with sublattices of $\mathbf{Z}^d$, that is, any lattice of the form $\bigoplus_{i\in I} b_i\mathbf{Z}$ for $(b_i)_{i\in I} \in (\mathbf{Z}^n)^I$ a linearly independent family of integral vectors. The algorithm works in the exact same manner for any integral scalar product, as remarked by Lovász and Scarf in [LS92]. For the sake of simplicity, the following exposition is conducted as in [LLL82] on full-rank sublattice of $\mathbf{Z}^d$.*

Algorithm 1: The original LLL algorithm.

```
Parameters: $\delta \in (1/4, 1)$
Input: Initial basis $(b_1, \dots, b_d)$
Result: A $\delta$ LLL-reduced basis
1  Compute the $\pi_i(b_i)$'s with the GSO process (Paragraph 2.1);
   // Size-reduce each vectors.
2  for $i = 2$ to $d$ do
3      for $j = i - 1$ to $d$ do
4          $b_i \leftarrow b_i - \left\lceil \frac{\langle b_i, \pi_i(b_j)\rangle}{\|\pi_i(b_i)\|^2} \right\rfloor \cdot b_j$;
5      end
6  end
   // Test if the Lovász condition is fulfilled, swap if not.
7  for $i = d - 1$ downto 1 do
8      if $\delta\|\pi_i(b_i)\|^2 > \left(\|\pi_{i+1}(b_{i+1})\|^2 + \frac{\langle b_{i+1}, \pi_i(b_i)\rangle}{\|\pi_i(b_i)\|^2}\right)$ then
9          Swap $b_i$ and $b_{i+1}$;
10         goto 1;
11     end
12 end
13 return $(b_1, \dots, b_d)$
```

2.3.1. *Decrease of the potential and complexity.* The soundness the algorithm is direct. Moreover, it terminates in polynomial time when $\delta < 1$. A classical argument relies on the study of the (square of the) product of the covolumes of the flag associated with a basis: $\prod_{i=1}^d \|\pi(b_i)\|^{2(n-i+1)}$, called *potential*. This value decreases by a factor at least δ^{-1} in each exchange step and is left unchanged by other operations. Indeed:

- Either a linear combination on b_k of the previous vectors is performed and the flag associated with the current basis is not altered.
- Or an exchange is done: assume without loss of generality that b_k and b_{k-1} are exchanged. Then, none of the $n-2$ products $\|\pi_1(b_1)\dots\pi_j(b_j)\|^2$ for $j \le k-1$ and $\|\pi_j(b_j)\dots\pi_j(b_d)\|^2$ for $j \ge k$ are modified. Since by construction the exchange occurs if $\|\pi_{k-1}(b_{k-1})\|^2$ shrinks by a factor at least δ^{-1}, then $\|\pi_1(b_1)\dots\pi_{k-1}(b_{k-1})\|^2$ also decreases by the same factor.

Thus the number of exchange steps—and consequently of iterations—is bounded by $\mathrm{O}\big(\log\|B\|_{\max} d^2\big)$ where B is the matrix of the initial basis.

As the cost of a loop iteration is of $O(dn)$ arithmetic operations on *rational* coefficients of length at most $\mathrm{O}(d\log\|B\|_{\max})$, the total cost in term of arithmetic operations is loosely bounded by $\mathrm{O}\big(d^6\log^3\|B\|_{\max}\big)$. By being more precise in the majoration of the bit-length of the integers appearing in LLL, this analysis can be improved, e.g. in [Kal83] where Kaltofen bounds the complexity by $\mathrm{O}\Big(\frac{d^5 \log^2 \|B\|_{\max}}{d+\log \|B\|_{\max}} M(d+\log \|B\|_{\max})\Big)$.

2.3.2. *A bound on the norm of reduced elements.*

Proposition 2.1. *Let $1/4 < \delta < 1$ be an admissible* LLL *parameter. Let $(b_1, \dots, b_d)$ a δ-*LLL *reduced basis of rank-d lattice $(\Lambda, \langle \cdot, \cdot \rangle)$. Then for any $1 \le k \le d$:*

$$\operatorname{covol}(b_1, \dots, b_k) \le \left(\delta - \frac{1}{4}\right)^{-\frac{(d-k)k}{4}} \operatorname{covol}(\Lambda)^{\frac{k}{d}}$$

Proof. Using the Lovász condition at index $1 \le i < d$, we have by orthogonality:

$$\|\pi_i(b_i)\|^2 \le \|\pi_i(b_{i+1})\|^2 = \|\pi_i(b_{i+1})\|^2 + \mu_{i,i+1}^2 \|\pi_i(b_i)\|^2$$

Using the size-reduction condition we then derive that:

$$\forall i \in \{1, \dots, d-1\}, \quad \|\pi_i(b_i)\|^2 \le \left(\delta - \frac{1}{4}\right)^{-1} \|\pi_{i+1}(b_{i+1})\|^2. \tag{3}$$

Let us denote by K the constant $\sqrt{\delta - \frac{1}{4}}^{-1}$ and ℓ_i the norm of the vector $\pi_i(b_i)$, so that inequalities of Equation (3) become:

$$\forall i \in \{1, \dots, d-1\}, \quad \ell_i \le K\ell_{i+1}.$$

Remark that $\operatorname{covol}(\Lambda) = \prod_{i=1}^d \ell_i$ by definition of the projections π_i. Hence we have:

$$\begin{aligned}
\operatorname{covol}(b_1, \dots, b_k)^d = \left(\prod_{i=1}^k \ell_i\right)^n &= \left(\prod_{i=1}^k \ell_i\right)^k \prod_{i=1}^k \underbrace{\ell_i^{d-k}}_{\le \ell_{k+1} \cdots \ell_n K^{\frac{(d+k+1-2i)(d-k)}{2}}} \\
&\le \left(\prod_{i=1}^k \ell_i\right)^k \left(\prod_{i=k+1}^d \ell_i\right)^k K^{\frac{1}{2}\sum_{i=1}^k (d+k+1-2i)(d-k)} \\
&= \operatorname{covol}(\Lambda)^k K^{\frac{d(d-k)k}{2}}
\end{aligned}$$

□

2.3.3. *Floating point representation.* The total cost of the LLL algorithm is dominated by the computation to handle the arithmetic with such representation of rational values, creating a bottleneck in the computation in the reduction. A first idea—introduced by De Weger in [Weg87]—to overcome this issue is to avoid the use of denominators by multiplying all the quantities involved by carefully chosen integers. While slightly more efficient in practice, the algorithm keeps the same asymptotic behavior. However, it is remarkable that the norm of these rational values remains small, and naturally leads to using approximations of the desired quantities, instead of computing with them in an exact manner. Translating directly the LLL algorithm with floating point approximations leads to severe drawbacks. First, the whole algorithm might not terminate, and even if it does, the output basis is not any longer guaranteed to be LLL-reduced.

The seminal *provable* floating-point version of the algorithm is due to Schnorr in [Sch88], leading to a complexity of $\mathrm{O}\big(d^4 \log(\|B\|_{\max}) M(d + \log \|B\|_{\max})\big)$. One of the key ingredient to achieve this reduction is to slightly relax the definition of the size-reduction, in order to compensate the approximation errors induced by the use of floating-point arithmetic:

Definition 2.3 ((δ, η)-LLL reduction)**.** *A basis $\mathcal{B}$ of a lattice is said to be (δ, η)-**LLL-reduced** for certain parameters $1/4 < \delta < 1$ and $1/2 < \eta < \sqrt{\delta}$ if the following conditions are satisfied:*

$$\forall i < j, \quad |\langle b_j, \pi_i(b_i)\rangle| \leq \eta \|\pi_i(b_i)\|^2 \quad \textit{(Size-Reduction condition)} \tag{4}$$

$$\forall i, \quad \delta\|\pi_i(b_i)\|^2 \leq \left(\|\pi_{i+1}(b_{i+1})\|^2 + \frac{\langle b_{i+1}, \pi_i(b_i)\rangle}{\|\pi_i(b_i)\|^2} \right) \quad \textit{(Lovász condition)} \tag{5}$$

Using naive multiplication, the cost of this is however still cubic in $\log(\|B\|_{\max})$. Number theory libraries and packages contain heuristic variants of the floating-point version of Schnorr and Euchner [SE94], like NTL [Sho17]. Numerous variants are built on this first version of Schnorr and Euchner, to eventually lead to the now widely used and fastest provable floating-point variant of Nguyen-Stehlé L^2 [NS09], implemented in the library FPLLL [ABC$^+$17].

2.4. The L^2 algorithm.

The L^2 algorithm is a variant of Schnorr-Euchner version of LLL [SE94]. By contrast with the original algorithm, L^2 computes the GSO coefficients on the fly as they are needed instead of doing a full orthogonalization at the start. It also uses a lazy size reduction inspired by the *Cholesky factorization algorithm.*

The Gram-Schmidt orthogonalization on $\mathcal{B}$ leads to the QR-decomposition of B into $B^* \cdot M$ where B^* is the matrix representing the $(\pi_i(b_i))_{1 \leq i \leq d}$, and M is the matrix of coefficients $M_{i,j} = \frac{\langle b_i, \pi_i(b_i)\rangle}{\|\pi_i(b_i)\|^2}$. Thus, by considering the Gram matrix associated to the basis[5], one gets:

$$G = M^T \cdot {B^*}^T \cdot B^* \cdot M = M^T D M$$

for a diagonal matrix D, since the rows B^* are by construction orthogonal. Denoting by R the matrix $D \cdot M$, we have:

$$G = R^T \cdot M = M^T \cdot R.$$

Hence, GSO coefficients are obtained by computing the $R_{i,j}$ inductively with the formula:

$$R_{i,j} = G_{i,j} - \sum_{k=1}^{j-1} M_{j,k} \cdot R_{i,k}.$$

Computing the quotient $\frac{R_{i,j}}{R_{j,j}}$ yields by construction $M_{i,j}$. Furthermore, the actual algorithm computes the quantities $s_j^{(i)} = \|b_i\|^2 - \sum_{k=1}^{j-1} M_{j,k} \cdot R_{i,k}$. for all $1 \leq j \leq i$, leading to a simple reformulation of the Lovász condition:

$$\delta \cdot R_{k-1,k-1} \leq s_{k-1}^{(k)}.$$

The Lazy Size-Reduction procedure is fully given in Algorithm 2 and the L^2 algorithm in Algorithm 3.

[5]Recall that $G = B^T B$ by definition.

These optimizations improve the running time of the lattice reduction to

$$\mathrm{O}\big(d^5(d+\log(\|B\|_{\max}))\log(\|B\|_{\max})\big),$$

while still being provable.

Algorithm 2: The lazy size reduction algorithm, LazyRed.

```
   Input: Initial basis $(b_1, \ldots, b_d)$
   Result: Basis $\mathcal{B}'$, transformation matrices $M$, $R$, $s$
 1 for $j = 1$ to $k - 1$ do
 2     $R_{i,j} \leftarrow G_{i,j}$;
 3     for $i = 1$ to $j - 1$ do
 4         $M_{i,j} \leftarrow R_{k,j}/R_{j,j}$;
 5     end
 6 end
 7 $s_1^{(k)} \leftarrow \|b_k\|^2$;
 8 for $j = 2$ to $k$ do
 9     $s_j^{(k)} \leftarrow s_{j-1}^{(k)} - M_{k,j-1} \cdot R_{k,j-1}$;
10 end
11 $R_{i,i} \leftarrow s_i^{(i)}$;
```

2.4.1. *Precision required.* The precision required by Algorithm 3 is

$$d\log\left(\frac{(1+\eta)^2}{(\delta-\eta)^2}+\epsilon\right)+o(d)$$

bits for any $\epsilon > 0$, i.e. almost linear in the dimension of the lattice. However, as presented in [NS06], it appears experimentally that, even though this bound is sharp, the number of bits required *on average* is lower.

Hence the challenge would be to manage to detect at run-time whether the precision is sufficient or not to soundly perform the computation. Then it would be possible to dynamically adapt the precision of the current computation and only work with a quasi-optimal precision. All of this can be achieved using *Interval Arithmetic.*

3. Interval Arithmetic and the certification property

3.1. **A primer on Interval Arithmetic.**

In this section, we present briefly Interval Arithmetic and focus on two types of representations of reals. For more details the interested reader can consult a more extensive reference, such as [Moo77].

3.1.1. *Interval Arithmetic in a nutshell.* Interval arithmetic is a representation of reals by intervals that contain them. For instance, one can specify that a value x is given with an error ϵ by considering the interval $[x-\epsilon, x+\epsilon]$, or manipulating the transcendent constant π with an error of at most 10^{-2} as the interval $[3.14, 3.15]$. Interval arithmetic is crucial in the context of *certified* numerical computations, where reals can only be represented with finite precision.

Algorithm 3: The L^2 Algorithm.

```
Parameters: $\delta \in (1/4, 1), \eta \in (1/2, \sqrt{\delta})$.
Input: Initial basis $(b_1, \ldots, b_d)$
Result: A $(\delta, \eta)$ LLL-reduced basis
1  Compute $G = G(b_1, \cdots, b_d)$ in exact integer arithmetic;
2  $R_{1,1} \leftarrow G_{1,1}$;
3  $k \leftarrow 2$; ;
4  while $k \leq d$ do
5      Apply length reduction LazyRed($k$);
6      $k' \leftarrow k$;
7      while $k \geq 2$ and $\delta R_{k-1,k-1} > s_{k-1}^{k'}$ do
8          $k \leftarrow k - 1$;
9      end
10     $R_{k,k} \leftarrow s_k^{k'}$;
11     if $k \neq k'$ then
12         for $i = 1$ to $k - 1$ do
13             $M_{k,i} \leftarrow M_{k',i}$;
14             $R_{k,i} \leftarrow R_{k',i}$;
15         end
16         Insert $b_{k'}$ at pos $k - 1$ and update matrices $M, R$;
17     end
18     $k \leftarrow k + 1$;
19 end
20 return $(b_1, \cdots, b_d)$
```

3.1.2. *Towards an algebra of Intervals.* In the following, we denote by $\underline{x}$ an interval, by $\underline{x}^-$—resp. $\underline{x}^+$—its lower value—resp. its greatest value—, that is to say: $\underline{x} = [\underline{x}^-, \underline{x}^+]$. Define its *diameter* as the positive real $\underline{x}^+ - \underline{x}^-$ and its *center* as the real $\frac{1}{2}(\underline{x}^+ + \underline{x}^-)$. We can now define abstractly an arithmetic on intervals:

Definition 3.1. *Let $\bowtie$ be a binary operation—resp. f be a function—over the reals, then the result $\underline{x} \bowtie \underline{y}$ of the operation between the intervals $\underline{x}$ and $\underline{y}$—resp $f(\underline{x})$, result of the application of f—is the smallest interval, in the sense of inclusion, containing*

$$\{x \bowtie y | (x, y) \in \underline{x} \times \underline{y}\} \qquad \text{—resp. } \{f(x) | x \in \underline{x}\}\text{—}.$$

For usual representations of reals, one can easily derive closed formulae for the arithmetic of the elementary operators. However, in the context of actual computations, requiring the equality in the above definition for arbitrary functions is in most case illusory, since reals cannot be represented exactly. Yet, only the inclusion of the result interval in the last defined sets is required to ensure the most desired property of interval arithmetic: *certification* of computations.

3.2. **Representations of reals.** We now present two different kind of representations of reals with intervals.

$$\begin{aligned}
\left[\underline{x}^-, \underline{x}^+\right] + \left[\underline{y}^-, \underline{y}^+\right] &= \left[\underline{x}^- +^- \underline{y}^-, \underline{x}^+ +^+ \underline{y}^+\right] \\
\left[\underline{x}^-, \underline{x}^+\right] - \left[\underline{y}^-, \underline{y}^+\right] &= \left[\underline{x}^- -^- \underline{y}^-, \underline{x}^+ -^+ \underline{y}^+\right] \\
\left[\underline{x}^-, \underline{x}^+\right] \times \left[\underline{y}^-, \underline{y}^+\right] &= \left[\min^-(\rho), \max^+(\rho)\right] \qquad \text{where} \quad \rho = \underline{x}^-\underline{y}^-, \underline{x}^+\underline{y}^-, \underline{x}^-\underline{y}^+, \underline{x}^+\underline{y}^+ \\
\left[\underline{x}^-, \underline{x}^+\right]^{-1} &= \left[\min^-(\frac{1}{\underline{x}^+}, \frac{1}{\underline{x}^-}), \max^+(\frac{1}{\underline{x}^+}, \frac{1}{\underline{x}^-})\right]
\end{aligned}$$

—$+^+$, $+^-$ *are here respectively the* $+$ *operator with higher approximation and with lower approximation. Same goes for the* $-^+, -^-, \min^-, \max^+$ *operators.*

FIGURE 1. Basic arithmetic operators in Interval Arithmetic

3.2.1. *Integral representation.* A naive yet convenient way to represent reals at finite precision is to use integers to represent the bounds of intervals. Specifically, this corresponds to consider integers made from the most-significant bits of the binary expansion of the desired reals. Formally, we have:

Definition 3.2 (Integral representation of reals)**.** *Let* $x \in \mathbf{R}$ *be an arbitrary real number and* $n > 0$ *a non-negative integer. Define its* integral representation at accuracy[6] n *as the symmetric interval centered on* $\lfloor 2^n x \rceil$ *of diameter* 1*, that is to say th interval:*

$$\underline{x}_n = \left[\lfloor 2^n x \rceil - \frac{1}{2}, \lfloor 2^n x \rceil + \frac{1}{2}\right]$$

Remark. *The interval* $\underline{x}_n$ *only needs to be represented as its center, since its bounds are simple translations of it, allowing thus a compact representation of the whole interval as a single integer. Notice that the space complexity of this representation is a function of the magnitude of the value* x *represented: it acts as an offset to the accuracy, its size being* $n + \lceil \log x \rceil$ *bits.*

3.2.2. *Interval Arithmetic and Floating-point approximations.* Another way to handle real values is to use floating point representation. Interval Arithmetic can indeed be used to handle exact reals by manipulating intervals whose bounds are represented by floating-point numbers at given precision. More precisely, if we denote by $\lfloor x \rfloor_n$ and $\lceil x \rceil_n$ respectively the largest floating-point number below x and the lowest floating-point number above x written with n bits, the tightest floating-point representation of x with n-bits of precision is the interval $I_n(x) = [\lfloor x \rfloor_n, \lceil x \rceil_n]$.

Explicitly, when approximating reals with floating point ends, basic arithmetic operations transpose directly into the formulae of the Figure 1.

3.3. **Certification property and inequality testing.** When real numbers are represented by intervals, the interval resulting from the evaluation of an algebraic expression *contains* the exact value of the evaluated expression. More precisely, for any family $(x_i)_{1 \leq i \leq n}$ of reals, and $(\underline{x}_i)_{1 \leq i \leq n}$ intervals such as for each i, $x_i \in \underline{x}_i$, then

$$f(x_1, \ldots, x_n) \in f(\underline{x}_1, \ldots, \underline{x}_N),$$

for any algebraic expression f. Therefore the results given in Interval Arithmetic are *certified.*

[6]We use here the denomination of "accuracy" instead of "precision" to avoid confusions with the floating-point precision as defined in paragraph 3.2.2.

For instance, and this is the main aspect we are using in the present work, one can compare in a certified manner a value δ with a real x, represented by $\underline{x}$ in Interval Arithmetic with floating-point representation at precision n:

- Either $\underline{x}^+ \le \lfloor \delta \rfloor_n$, which certifies that $x \le \delta$.
- Either $\underline{x}^- \ge \lceil \delta \rceil_n$, which certifies that $x \ge \delta$.
- Or $\delta \in \underline{x}$, and then the precision chosen is not sufficient to conclude.

If this test fails, one is only informed that the interval $\underline{x}$ is too large to certify the desired property, allowing to conclude that the precision chosen is not sufficient to assert the inequality. This disjunction is easily adapted to the integral representation setting.

In short, Interval Arithmetic used in such a way allows detecting a lack of precision or accuracy at runtime of a numerical algorithm.

4. Approximate lattices

4.1. **Matrix representation and positive-definiteness.**

4.1.1. *Integral representation of matrices.* The Interval Arithmetic framework can be extended to represent real-valued matrices, in particular with the integral representation presented in Definition 3.2.

Definition 4.1 (Matrix integral representation)**.** *Let $A = (a_{i,j})_{i,j} \in \mathbf{R}^{d\times d}$ an arbitrary real matrix of size d and $n > 0$ be a fixed positive integer. The matrix of intervals*

$$\underline{A}_n = (\underline{a_{i,j}}_n)_{1\le i,j\le d},$$

is said to integrally represent A at accuracy n*.*

We may omit the subscript n when the accuracy is clear from the context. Let us broaden the meaning of the $\in$ inclusion operator to matrix representation as follows: for a matrix A, define $B \in \underline{A}_n$ as being $\exists \Delta \in [-\frac{1}{2}, \frac{1}{2}]^{d\times d}$, $B = \lfloor 2^n A \rceil + \Delta$. Since in essence, the integral representation relies on a scale of coefficients, when studying matrix representations we are interested in sets of matrices which remain invariant under scaling, that is under any linear operator of the form $M \mapsto \alpha M$, for $\alpha > 0$ a non-negative real. For instance the set of non-singular (resp. positive-definite) matrices over $\mathbf{R}$ shares this property by multilinearity of the determinant (resp. by multiplicativity of the eigenvalue map). In order to approximately represent lattices within the framework of interval arithmetic, we aim at representing the inner product of lattices. This boils down to represent positive-definite real matrices. The systematic treatment of this question is the goal of next section.

4.1.2. *Minimal representation of positive-definite matrices.* Let us fix d a non-negative integer and $A \in \mathcal{S}_d^{++}(\mathbf{R})$ a symmetric positive-definite matrix. As presented in Section 3.3 when representing a positive real x by an interval $\underline{x}$, we can only assert that x is positive if every element of the interval $\underline{x}$ is positive. The natural extension of this property to positive-definite matrices is encoded in the following definition:

Definition 4.2 (Positive-definite representation)**.** *Let n be a fixed positive integer. Let $S \in \mathcal{S}_d^{++}(\mathbf{R})$ a symmetric positive-definite matrix. A representation $\underline{S}_n$ of S is positive-definite if every symmetric matrix $S' \in \underline{S}_n$ is positive-definite.*

Clearly, if a representation of A is positive-definite at accuracy n, then it is also the case for any representation at greater accuracy $n' \geq n$. Hence, given a fixed matrix, a natural question is to determine the minimal accuracy n_0 needed to obtain a positive-definite representation:

Lemma 4.1. *Let $S = (s_{i,j})_{i,j} \in S_d^{++}(\mathbf{R})$ a symmetric positive-definite matrix of size d, then the minimal accuracy required to get a positive-definite representation is bounded as follow:*

$$\max\left(\left\lfloor \frac{1}{2}\log d - \log \lambda_d(S) \right\rfloor, 0\right) \leq n_0 \leq \max(\lceil \log d - \log \lambda_d(S) \rceil, 1)$$

where $\lambda_d(S)$ is the least eigenvalue of S.

Proof. Let $n \in \mathbf{N}$ and S a symmetric positive-definite.

- *Lower bound*: Let S' be a symmetric matrix in $\bar{S}_n$. By definition there exists Δ' a symmetric matrix with all entries in the real interval $\left[-\frac{1}{2}, \frac{1}{2}\right]$, such that $S' = \lfloor 2^n S \rceil + \Delta'$. Moreover there exists Δ'' a symmetric matrix with all coefficients in $[-\frac{1}{2}, \frac{1}{2}]$, such that $\lfloor 2^n S \rceil = 2^n S + \Delta''$. Now write $\Delta = (\delta_{i,j})$ for the sum $\Delta' + \Delta''$ of these two perturbations matrices; hence $\|\Delta\|_{\max} \leq 1$ by construction. By virtue of Spectral Theorem (see [HJ12] for a comprehensive reference), the real symmetric matrices S' and Δ are diagonalizable with real spectrum, so that one can consider their respective least eigenvalue $\lambda_d(S')$ and $\lambda_d(\Delta)$. By Weyl's first inequality (See [Wey12]), $\lambda_d(S') \geq 2^n \lambda_d(S) + \lambda_d(\Delta)$. Thus if $2^n > -\frac{\lambda_d(\Delta)}{\lambda_d(S)}$ then S' is positive-definite. Denoting by Δ_i the columns of Δ we have by Hadamard's inequality:
$$\det \Delta \leq \prod_{i=1}^{d} \|\Delta_i\|_2 \leq \left(\max_{1\leq i\leq d} \sqrt{\sum_{j=1}^{d} |\delta_{i,j}|^2}\right)^d \leq d^{\frac{d}{2}},$$
yielding $|\lambda_d(\Delta)| \leq \sqrt{d}$. Plugging back in the previous inequality and taking the logarithm recovers the left hand side of the announced result.
- *Upper bound*: Let $\Delta = (-1)_{1\leq i,j\leq d}$. A simple calculation ensures that $X^d + dX^{d-1} = X^{d-1}(X + d)$ is the characteristic polynomial of Δ, and therefore that $\lambda_1(\Delta) = -d$. Notice that we have from Weyl's second inequality (See [Wey12]):
$$\lambda_d(S + \Delta) \leq 2^n \lambda_d(S) + \lambda_1(\Delta) = 2^n \lambda_d(S) - d,$$
Hence if $2^n \lambda(S) < d$, then $\lambda_d(S + \Delta)$ is negative, so that $S + \Delta$ is not definite positive. Since $(S + \Delta) \in \underline{S}_n$, this implies that the interval matrix $\underline{S}_n$ is not positive definite. As before, we just have to take the logarithm to conclude.

□

4.2. **Representation of lattices.** In all of the following we fix a lattice $(\Gamma, \langle \cdot, \cdot \rangle)$ of rank d, its ambient space $V = (\Gamma \otimes \mathbf{Q}, \langle \cdot, \cdot \rangle)$, as well as $\gamma = (\gamma_1, \ldots, \gamma_d)$ one of its basis.

Now consider a rank $r \leq d$ sublattice $\Lambda \subset \Gamma$ represented by a generating family $\ell = (\ell_1, \ldots, \ell_p)$. Therefore the triple (V, γ, ℓ) univoquely describe the lattice Λ. Remark that ℓ can be encoded as a $d \times p$ integral matrix L, which is its matricial representation in the basis γ, and the inner product $\langle \cdot, \cdot \rangle$ by the $d \times d$ Gram-matrix

$\mathcal{G}_\gamma = (\langle \gamma_i, \gamma_j \rangle)_{1 \le i,j \le d}$. This last matrix is *a priori* real-valued and thus, must be approximated at finite precision to be computationally handled, for instance with the integral representation introduced in Section 4.1.1. All of these considerations lead to the following definition:

Definition 4.3 (Approximate representation of a lattice)**.** *Let $\mathcal{G}_\gamma$ and L as above and n a non-negative integer. Denote by G the matrix of centers of the integral representation $\underline{\mathcal{G}_\gamma}_n$ at accuracy n of the Gram-matrix $\mathcal{G}_\gamma$. Then the pair $(G, L) \in \mathbf{Z}^{d\times d} \times \mathbf{Z}^{d\times p}$ of integral matrices is said to* represent at accuracy n *the lattice Λ in the basis γ of Γ.*

4.2.1. *Computation of the inner product.* By the certification property of the Interval Arithmetic, the computation of the scalar product of two vectors $a, b \in \Lambda$ from the interval matrix represented by the matrix G is yields intervals containing its exact value $\langle a, b\rangle$. Formally let us consider the (column) vectors of integers A and B representing respectively the elements $a, b \in \Lambda$ in the basis γ. Whence, the inner product $\langle a, b\rangle$ is by definition of $\mathcal{G}_\gamma$ and G:

$$\langle a, b\rangle = \frac{1}{2^n}\left(A^T G B + \underbrace{A^T \cdot [2^n \mathcal{G}_\gamma - G] \cdot B}_{\le \frac{1}{2}(\sum_i |A_i|)(\sum_i |B_i|)} \right), \tag{6}$$

the inequality resulting from the bound $\|2^n\mathcal{G}_\gamma - G\|_{\max} \le 1$. Equation (6) actually states that the value of the (scaled) inner product $2^n\langle a, b\rangle$ is contained in the interval $\langle a, b\rangle_n$ centered on the integer $A^T \cdot \lfloor 2^n G \rceil \cdot B$ of diameter $2^{-n-1}(\sum_i |A_i|)(\sum_i |B_i|)$.

4.2.2. *Proper representation.* A lattice is by definition discrete in its ambient space; in particular, the infimum $\lambda_1(\Lambda)$ of the norm of non-zero elements in the lattice is attained and is strictly positive. By analogy with sublattices of $\mathbf{Z}^d$, it is technically convenient to re-scale the original inner product of the lattice to impose $\lambda_1(\Lambda) \ge 1$. The encoding of this property in the interval framework is imposing the interval $\|x\|_n = \sqrt{\langle x, x\rangle_n}$ representing the norm of a non-zero vector $x \in \Lambda$ to be greater than 1, as an interval. By Equation (6) this condition is equivalent to $\underline{\mathcal{G}_\gamma}_n - 2^{-n}\mathbf{I_d}$ being positive definite. Such a representation is said to be *proper*:

Definition 4.4 (*Approximate* proper representation of a lattice)**.** *Let (G, L) be a pair of integral matrices representing at accuracy n a lattice Λ in a basis γ. This representation is said to be proper if for any non zero vector $x \in \Lambda$, $\|x\|_n > 1$, that is if $\underline{\mathcal{G}_\gamma}_n - 2^{-n}\mathbf{I_d}$ is positive definite as an integer interval matrix (see Definition 4.3).*

The characterization of Lemma 4.1 extends naturally into the following bounds for proper representation.

Proposition 4.1. *The minimal accuracy n_0 needed for a representation (G, L) representation of the lattice $(\Lambda, \langle\cdot,\cdot\rangle)$ satisfies:*

$$\max\Big(\Big\lfloor \log(1+\sqrt{d}) - \log\lambda_d(S_{\mathcal{B}})\Big\rfloor, 0\Big) \le n_0 \le \max(\lceil \log(1+d) - \log\lambda_d(S_{\mathcal{B}})\rceil, 1)$$

4.3. **Lattice reduction as a computational problem.** Suppose now that the Gram-matrix $\mathcal{G}_\gamma = (\langle \gamma_i, \gamma_j\rangle)_{1\le i,j\le d}$ representing the inner product of the ambient space $\Gamma \otimes_{\mathbf{Z}} \mathbf{Q}$ is unknown but that we are given an oracle $\mathcal{O}_\gamma$ that can computes its representation at any accuracy.

The computational problems associated with reduction theory can be then written in the exposed framework as:

Problem (Lattice Reduction for proper reprentation)**.** *Let δ, η be admissible* LLL *parameters. Given $L \in \mathbf{Z}^{p\times d}$ encoding a sublattice Λ in the basis γ of Γ, as well as the oracle $\mathcal{O}_\gamma$, find a basis represented by $L' \in \mathbf{Z}^{d\times d}$ in γ, which is a (δ, η)-*LLL *reduced basis of Λ.*

4.3.1. *Accuracy of representation and space complexity.* Let (G, L) represents Λ at accuracy $n \in \mathbf{N}^*$ in the basis γ, then the magnitude of the coefficients of G is 2^n times the magnitude of the coefficients of $\mathcal{G}_\gamma$, and thus G needs at most $\mathrm{O}\big(d^2(n + \log \|\mathcal{G}_\gamma\|_{\max})\big)$ bits to be encoded. bits. Suppose that the matrix $\mathcal{G}_\gamma$ satisfies $\ell = \lambda_d(\mathcal{G}_\gamma) > d$—implying that its determinant is exponentially large w.r.t the dimension, being at least equal to $2^{d\log(d)}$—, then the minimal accuracy required to get a proper representation is at most 1 by Proposition 4.1. Hence, the space occupation of the representation of S is a $\mathrm{O}\big(d^2 \log \|\mathcal{G}_\gamma\|_{\max}\big)$. But we can do slightly better by taking advantage of the magnitude of ℓ. Indeed the re-scaled matrix $d\ell^{-1}\mathcal{G}_\gamma$ has now d for least eigenvalue. Notice then that by Proposition 4.1 the minimal accuracy required to get a proper representation is still at most 1, but the space complexity collapses to $\mathrm{O}\big(d^2(\log \|\mathcal{G}_\gamma\|_{\max} - \log \ell + \log d)\big)$. This space optimization also induces an improved running-time for the reduction algorithm, its complexity being a function of the magnitude of the inner product representation[7], as presented in Section 5.3.

5. Generalized LLL reduction with Interval Arithmetic

5.1. **Interval Arithmetic L^2 reduction.** Like previously, we fix a lattice $(\Gamma, \langle \cdot, \cdot \rangle)$ of rank d and $\gamma = (\gamma_1, \ldots, \gamma_d)$ one of its a basis. Let (G, L) be a *proper representation* of a (sub)lattice $\Lambda \subseteq \Gamma$ at accuracy n in γ.

5.1.1. *Using Interval Arithmetic in* LLL. In order to benefit from the certification property evoked in Section 3.3, we use interval representation—with floating-points bounds—for handling floating-point variables appearing in the L^2 algorithm. An internal precision p_0 is chosen to represent the bounds of these intervals. Modifying the L^2 algorithm in such a way implies to transform the conditional statements, which are inequalities between floating-point values, by the tests over intervals described in Section 3.3. It is then possible to detect *at runtime* whether this choice of precision was sufficient to achieve a provable reduction. Indeed, in the case where the Lovász condition test detects a precision defect, the reduction stops and yields an error. Hence, we change the code from line 7 to 9 in Algorithm 3 to handle this possibility, where a test between two intervals $\underline{a}$ and $\underline{b}$, $(\underline{a} > \underline{b})$ returns either a Boolean value if the result can be certified thanks to Interval Arithmetic, either $\perp$ if not. Notice that *in practice*, in order to optimize the update of the Gram-matrix (and related quantities) of the current basis after unimodular operations, we do not recompute it entirely from scratch. On the contrary, we only update the coefficients by applying matrix congruence. The drawback of this faster approach is to amplify the diameter of intervals at each iteration. Hence, if a precision error is detected we replay the last iteration of the reduction with a freshly computed

[7]Computationally speaking, this corresponds to perform the reduction on only the most significant bits of $\mathcal{G}_\gamma$.

Gram-matrix, which contains the least error possible with regards to the available data. If an error is still detected, the algorithm eventually yields a precision error. We name this modified procedure $\overline{\text{LL}}$ and fully present it in Algorithm 4.

5.1.2. *Soundness and internal precision.* One naturally wonders if the precision needed by the reduction procedure $\overline{\text{LL}}$ to correctly run can be bounded independently of the coefficients appearing in the matrices G and L. This is the case thanks to the properness of the representation and one can even derive an explicit estimate of the required precision :

Theorem 5.1. *Let* $(\Lambda, \langle\cdot,\cdot\rangle)$ *a rank* d *lattice,* properly *represented at accuracy* n *by the pair* (G, L)*. The* $\overline{\text{LL}}$*(Algorithm 5) at most*

$$T(d,\delta,\eta,\epsilon) = \left\lceil d\log\left[d^{\frac{2}{d}}\frac{(1+\eta)^2+\epsilon}{\epsilon(\delta-\eta^2)}\right] + 10\right\rceil$$

bits to soundly reduce the lattice Λ*, for any* $\epsilon > 0$*, where* $\epsilon = \min\bigl(\eta - \frac{1}{2}, 1-\delta\bigr)$ *the minimum distance of the chosen* LLL *parameters to the (theoretical) parameters* $\frac{1}{2}$ *and* 1*.*

For usual choices of parameters—δ close to 1 and η close to 1/2—the bound $T(d,\delta,\eta,\epsilon)$ of Section 5.1.2 reduces roughly to $1.6d + o(d)$. This last bound is tight in the sense that some lattices require such a complexity to be reduced by ADAPTIVE-LLL. However, in [NS06], Nguyen, and Stehlé presented an experimental heuristic on the precision required by L^2 to safely perform its computation. This heuristic is still experimentally valid when reducing arbitrary real lattices with ADAPTIVE-LLL:

Heuristic 5.1 (Adapted from [NS06])**.** *For most lattices, given as an arbitrary generating family, a precision of* $0.25d + o(d)$ *bits for floating-point calculations is sufficient for the* ADAPTIVE-LLL *to run correctly.*

5.2. **Towards adaptive precision and accuracy.**

5.2.1. *Adaptive precision.* Since by construction the $\overline{\text{LL}}$ Algorithm allows detecting if the choice of internal precision p_0 was sufficient to soundly reduce the lattice Λ, one can naturally wrap this procedure in a loop that double the precision p_0 each time an error is caught. This yields an *adaptive precision* reduction algorithm. Depending on the internal representation used, the cost of Interval Arithmetic with floating-point representation of bounds is up to 4 times the cost of classical large-precision floating-point arithmetic. However, since the complexity of floating-point multiplication is superlinear and the precision growth is by design geometric, the total complexity of the ADAPTIVE-LLL is asymptotically dominated by the reduction performed at the maximal precision[8].

5.2.2. *Adaptive accuracy.* Let us recall that the lattice $(\Gamma, \langle\cdot,\cdot\rangle)$ of rank d and $\gamma = (\gamma_1, \ldots, \gamma_d)$ one of its a basis are fixed. Suppose that an oracle $\mathcal{O}_\gamma$ that outputs an integral representation of the Gram-matrix $\mathcal{G}_\gamma = (\langle\gamma_i, \gamma_j\rangle)_{1\le i,j\le d}$ at arbitrary accuracy is also given. We consider a sublattice $\Lambda \subseteq \Gamma$, described by $L \in \mathbf{Z}^{p\times d}$ in γ and aim at solving the lattice reduction problem, stated in Section 4.3. Let us choose n_0 an arbitrary accuracy and denote by $G_{n_0} = \mathcal{O}_\gamma(n_0)$ the output of the

[8]In practice, for lattices of rank few hundreds it appears nonetheless that the computational cost of the first iterations lies between 20% and 40% of the total cost.

Algorithm 4: The $\overline{\text{LL}}$ Algorithm.

```
Parameters: $\delta \in (1/4, 1), \eta \in (1/2, \sqrt{\delta})$ admissible LLL parameters, $p_0 \in \mathbf{N}$
            the internal precision used for floating-point representation.
Input: $\gamma$ a basis of a lattice $(\Gamma, \langle \cdot, \cdot \rangle)$.
Input: An approximate representation $(G, L)$ of a sublattice $\Lambda \subset \Gamma$ at
       accuracy $n$ in the basis $\gamma$.
Result: A $(\delta, \eta)$ LLL-reduced basis represented as $L' \in \mathcal{M}_n(\mathbf{Z})$.
1  $k \leftarrow 2$ ;
   // Compute the Gram matrix of the basis represented by $L$, as in
      Section 4.2.1.
2  Compute $\underline{GramL} \leftarrow \left(\left[L_i^T G L_j - \|L_i\|_1 \|L_j\|_1 ; L_i^T G L_j + \|L_i\|_1 \|L_j\|_1\right]\right)_{1 \le i,j \le d}$ ;
3  fresh $\leftarrow$ true;
4  $\underline{R_{1,1}} \leftarrow \underline{GramL_{1,1}}$;
5  while $k \le d$ do
6      Apply size-reduction LazyRed($k$);
7      $k' \leftarrow k$;
8      while $k \ge 2$ do
9          ret $\leftarrow (\underline{\delta} \cdot \underline{R_{k-1,k-1}} > \underline{s_{k-1}^{k'}})$;
10         if $ret$ = true then
11             $k \leftarrow k - 1$;
12         else if $ret$ = false then
13             break;
14         else
15             if $fresh$ = true then
16                 return ErrorPrecision;
17             else
                   // Force the re-computation of the inner products
                      directly from $S_{\mathcal{B}}$.
18                 goto 2;
19             end
20         end
21     $\underline{R_{k,k}} \leftarrow \underline{s_k^{k'}}$;
22     if $k \neq k'$ then
23         for $i = 1$ to $k - 1$ do
24             $M_{k,i} \leftarrow M_{k',i}$;
25             $\underline{R_{k,i}} \leftarrow \underline{R_{k',i}}$;
26         end
27         Insert $L_{k'}$ at position $k - 1$ in $L$;
           // Done by applying the corresponding matrix unimodular
              transformation.
28         Update matrices $\underline{M}, \underline{R}$;
29         fresh $\leftarrow$ false;
30     end
31     $k \leftarrow k + 1$;
32 end
33 return $(L)$
```

oracle when feed with n_0. By definition this means that (G_{n_0}, L) is a representation of Λ in γ at accuracy n_0, even thought non necessarily proper. Suppose now that we launch the adaptive precision reduction (Section 5.2.1) on the pair (G_{n_0}, L). Two outcomes are possible:

- Either the precision goes beyond the theoretical bound $T(d, \delta, \eta, \epsilon)$, meaning that the representation (G, L) is not proper by virtue of Theorem 5.1.
- Either the reduction terminates[9].

Since we don't know *a priori* the accuracy needed to obtain a proper representation of $\mathcal{G}_\gamma$, the last case analysis leads to the same design principle as for the precision: wrapping the whole reduction procedure in a loop that double the accuracy each time an error is raised. The full outline of this strategy is exposed in Algorithm 5.

Algorithm 5: The ADAPTIVE-LLL algorithm.

Parameters: $\delta \in (1/4, 1), \eta \in (1/2, \sqrt{\delta}.)$, $p_0 \in \mathbf{N}$ initial precision of the algorithm for floating-point representation, n_0 initial accuracy for representing the scalar product.
Input: γ a basis of a lattice $(\Gamma, \langle \cdot, \cdot \rangle)$, and $\mathcal{O}_\gamma(\mathrm{n})$ an oracle that compute the integral representation of the inner product $\langle \cdot, \cdot \rangle$ at accuracy n.
Input: A generating family represented by L in γ of a sublattice $\Lambda \subset \Gamma$.
Result: A (δ, η) LLL-reduced basis of Λ represented as $L' \in \mathbf{Z}^{\mathrm{rk}(\Lambda)\times \mathrm{rk}(\Lambda)}$.

```
1  prec ← $p_0$;
2  accu ← $n_0$;
3  $G \leftarrow \mathcal{O}_\gamma$(accu);
4  succeed ← false;
5  repeat
6      prec ← prec×2;
7      if $\overline{\mathrm{LL}}(G, L) \neq$ ErrorPrecision then
8          succeed ← true;
9      end
       // T$(d, \delta, \eta)$ is the theoretical bound of Theorem 5.1
10     if prec $> T(d, \delta, \eta, \min(\eta - \frac{1}{2}, 1 - \delta))$ then
           // Not enough accuracy to represent the inner product
11         accu ← accu×2;
12         $G \leftarrow \mathcal{O}_\gamma$(accu);
13     end
14 until succeed = true;
15 return $(L)$
```

5.3. **Remarks on time complexity of ADAPTIVE-LLL.** The ADAPTIVE-LLL algorithm terminates since there exists a minimal accuracy at which the representation of the lattice becomes proper by Theorem 5.1. We can be more precise and estimate the complexity of the full reduction:

[9]This means that the representation is in fact proper.

Theorem 5.2. *Let $(\Lambda, \langle\cdot,\cdot\rangle)$ a rank d lattice represented by a generating family L in a basis $\gamma \in \Gamma^d$. Let n_0 be the minimal accuracy needed on representations of Λ to be proper. Then:*

- *The* ADAPTIVE-LLL *(Algorithm 5) operates in at most*

$$O\big(d^3(\log\|B\|_{max} + n_0 + \log\|S\|_{max})(n_0 + d + \log\|B\|_{max} + \log\|L\|_{max})\mathcal{M}(d) + \log(n)C_{\mathcal{O}_\gamma}(n_0)\big)$$

  arithmetic operations, where $C_{\mathcal{O}_\gamma}(n_0)$ is the complexity of one call of the oracle $\mathcal{O}_\gamma$ on the input n_0.
- *When the scalar product $\langle\cdot,\cdot\rangle$ is integral, the previous complexity reduces to:*

$$O\big(d^3(\log\|B\|_{max} + \log\|S\|_{max})(d + \log\|B\|_{max} + \log\|S\|_{max})\mathcal{M}(d)\big)$$

  arithmetic operations.

In the case where we are using the identity matrix as inner product[10]. Under Heuristic 5.1, the ADAPTIVE-LLL algorithm gains a constant factor on the proved version of L^2, corresponding to the gap between mandatory precision on the average and on the worst-case, while still being a provable reduction.

6. Application in Algebraic Number Theory: towards computational Minkovsky theory.

Let recall some basic facts on algebraic number theory to point out the emergence of a natural class of lattices, the so-called *ideal lattices.*

6.1. Number Fields.

Let $\mathbf{K} = \mathbf{Q}(\alpha)$ be a number field of dimension d, then there exists a monic irreducible degree-d polynomial $P \in \mathbf{Z}[X]$ such that $\mathbf{K} \cong \mathbf{Q}[X]/(P)$ and $P(\alpha) = 0$. Denoting by $(\alpha_1, \dots, \alpha_d) \in \mathbf{C}^n$ its distinct complex roots, each embedding—i.e. field homomorphism—$\sigma_i : \mathbf{K} \to \mathbf{C}$ is the evaluation of $a \in \mathbf{K}$, viewed as a polynomial modulo P, at the root α_i, that is $\sigma_i : a \mapsto a(\alpha_i)$. Classically, embeddings arising from the real (resp. complex) roots are called real (resp. complex) embeddings. With r_1 real roots and r_2 pairs of complex roots—$d = r_1 + 2r_2$ —, we have $\mathbf{K} \otimes_{\mathbf{Q}} \mathbf{R} \cong \mathbf{R}^{r_1} \times \mathbf{C}^{r_2} \cong \mathbf{R}^d$, the isomorphism being given by the *Archimedean embedding*, defined as:

$$\sigma : \left| \begin{array}{ccccc} \mathbf{K} \otimes_{\mathbf{Q}} \mathbf{R} & \longrightarrow & \mathbf{R}^{r_1} \times \mathbf{C}^{r_2} & \longrightarrow & \mathbf{R}^d \\ x & \longmapsto & \big(\underbrace{\sigma_1(x) \dots \sigma_{r_1}(x)}_{r}, \underbrace{\sigma_{r_1+1}(x) \dots \sigma_{r_1+r_2}(x)}_{c}\big)^T & \longmapsto & \big(r, \sqrt{2}\Re(c), \sqrt{2}\Im(c)\big)^T \end{array} \right.$$

where $\sigma_1, \dots, \sigma_{r_1}$ are the real embeddings and $\sigma_{r_1+1}, \dots, \sigma_n$ are the complex embeddings for which σ_{r_1+j} is paired with its complex conjugate $\sigma_{r_1+r_2+j}$. The number field $\mathbf{K}$ is then viewed as an Euclidean $\mathbf{R}$-vector space endowed with the inner product

$$\langle a, b\rangle_\sigma = \sum_{\sigma:\mathbf{K}\to\mathbf{C}} \sigma(a)\overline{\sigma(b)}$$

where σ ranges over all the $r_1 + 2r_2$ embeddings $\mathbf{K} \to \mathbf{C}$. Whence, we can derive from this inner product a Euclidean norm $\|\cdot\|$ over $\mathbf{K}$.

6.2. Ring of integers, integer ideals.

[10]That is reducing so-called integral Euclidean lattices.

6.2.1. *Integers, orders, and ideals.* An element γ of $\mathbf{K}$ is said to be *integral* if its minimal polynomial has integral coefficients and is monic. The *ring of integers*, $\mathfrak{o}_{\mathbf{K}}$, or *maximal order*, of $\mathbf{K}$ is the ring of all integral elements contained in $\mathbf{K}$. More generally a free $\mathbf{Z}$-module $\mathfrak{o}$ embedded in $\mathbf{K}$, such as $\mathbf{Q}\mathfrak{o} = \mathbf{K}$ is called an order of $\mathbf{K}$, and necessarily lies in $\mathfrak{o}_{\mathbf{K}}$. As a finite-rank sub-module of the field $\mathbf{K}$, there exists a finite family $(\gamma_i)_{i\in I}$, called an— integral—*basis* of the order, such that

$$\mathfrak{o} \cong \bigoplus_{i\in I} \gamma_i \mathbf{Z}.$$

An additive subgroup $\mathfrak{a}$ of $\mathfrak{o}$ for which the coset $a \cdot \mathfrak{o} = \{a \cdot x | x \in \mathfrak{o}\}$ lies in $\mathfrak{a}$ for every $a \in \mathfrak{a}$, is called an *ideal* of the number field.

6.2.2. *Canonical embedding and ideals.* The Archimedean embedding endows any integral ideal $\mathfrak{a}$ of an order $\mathfrak{o}$ with a lattice structure, namely as $(\mathfrak{a}, \langle \cdot, \cdot \rangle_\sigma)$.

In particular, any order $(\mathfrak{o}, \langle \cdot, \cdot \rangle_\sigma))$ is also a lattice. The square of its (co)volume, denoted by $\Delta_{\mathfrak{o}}$, is called its *discriminant.* Therefore, one can compute the discriminant as a determinant: for $\gamma = (\gamma_1, \ldots, \gamma_n)$ an integral basis of $\mathfrak{o}$, we have

$$\Delta_{\mathfrak{o}} = \det(\mathcal{G}_\gamma) = \left| \det \begin{pmatrix} \sigma_1(\gamma_1) & \sigma_1(\gamma_2) & \cdots & \sigma_1(\gamma_n) \\ \sigma_2(\gamma_1) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \sigma_n(\gamma_1) & \cdots & \cdots & \sigma_n(\gamma_n) \end{pmatrix} \right|^2,$$

where $\mathcal{G}_\gamma = \left(\sum_\sigma \sigma(\gamma_i)\overline{\sigma(\gamma_j)} \right)_{i,j}$ is the Gram-matrix associated to γ.

Specifically, the discriminant of the maximal order, denoted by $\Delta_{\mathbf{K}}$ is called discriminant of the field. Loosely speaking, the discriminant is a size measurement of the order. This is one of the reason it is used to express the complexity when working with number fields or rings of integers.

6.2.3. *Equation orders.* An integer $\theta \in \mathbf{K}$ defines univoquely an order $\mathbf{Z}[\theta] \subseteq \mathfrak{o}_{\mathbf{K}}$, called *equation order* defined by θ, where

$$\mathbf{Z}[\theta] = \{R(\theta) \mid R \in \mathbf{Z}[X]\} = \bigoplus_{i=0}^{\delta-1} \theta^i \mathbf{Z},$$

with δ the degree of its minimal polynomial. In particular, when considering α a root of a defining polynomial f of $\mathbf{K}$, the order $\mathbf{Z}[\alpha]$ is full-rank in $\mathfrak{o}_{\mathbf{K}}$, a *natural* basis of this order being the power-basis $A = (1, \alpha, \ldots, \alpha^{d-1})$. Such an order is called a *principal equation order* of $\mathbf{K}$. By definition of the Archimedean embedding, the Gram-matrix $\mathcal{G}_A$ associated to $\mathbf{Z}[\alpha]$ in A is the product of the Vandermonde[11] matrix $V_\sigma = V(\sigma_1(X), \cdots, \sigma_d(X))$ by its conjugate-transpose. Its general term is then:

$$V_\sigma \cdot V_\sigma^\dagger = \left(\sum_{k=1}^{d} \alpha_k^i \overline{\alpha_k}^j \right)_{1 \le i \le j \le d}.$$

[11]We recall that the Vandermonde matrix is a matrix with the terms of a geometric progression in each column, i.e., an $m \times n$ matrix of general term

$$V_{i,j} = \alpha_j^{i-1}.$$

Algorithm 6: The inner product representation oracle.

```
Input: A monic irreducible polynomial $P \in \mathbf{Z}[X]$.
Input: A basis $\gamma = (\gamma_1, \cdots, \gamma_n)$ of an order $\mathfrak{o}$ in a number field
       $\mathbf{K} \cong \mathbf{Q}[X]/(P)$.
Input: A non negative integer $n$.
Result: An integral representation at precision $n$ of $\mathcal{G}_\gamma = (\langle \gamma_i, \gamma_j \rangle)_{i,j}$.
1  precRoots $\leftarrow n_0$; $d \leftarrow \deg P$;
   // (Re)compute the canonical embeddings.
2  $(\alpha_1, \dots, \alpha_d) \leftarrow$ FindRoots(P, precRoots);
3  for $i = 1 \dots d$ do
4      for $j = 1 \dots d$ do
5          $\sigma_i(\gamma_j) \leftarrow \gamma_j(\alpha_i)$;
6      end
7  end
   // Derive the inner product matrix in $\gamma$.
8  for $i = 1 \dots d$ do
9      for $j = 1 \dots d$ do
10         $G \leftarrow \sum_{k=1}^{d} \sigma_k(\gamma_i)\overline{\sigma_j(\gamma_j)}$;
11     end
12 end
13 return $\lfloor 2^n \times G \rceil$;
```

6.3. **Practical reduction theory for ideals.** In all of the following we fix $\mathbf{K} = \mathbf{Q}(\alpha) \cong \mathbf{Q}(X)/(P)$ a number field and $\mathfrak{o}$ one of its order, as well as $\gamma = (\gamma_1, \dots, \gamma_d)$ a basis. The scalar product of the corresponding lattice is then naturally the matrix $\mathcal{G}_\gamma$ above-defined. Now consider an ideal $\mathfrak{a} \subseteq \mathfrak{o}$, presented by a generating family $(a_1, \dots, a_r)$ as $\mathbf{Z}$-module. Decomposing each vector a_i in the basis γ yields a matrix $A \in \mathbf{Z}^{d \times r}$.

This is precisely the setting of the reduction problem described in Section 4.3. In order to run the ADAPTIVE-LLL procedure described in Section 5.1, we need to compute the matrix $\mathcal{G}_\gamma$ at sufficient precision, in order to derive a proper representation of the lattice. This falls back to the computation of the values of the Archimedean embedding on the basis elements γ_i, which is a simple polynomial evaluation over the set of complex roots of the defining polynomial P of $\mathbf{K}$. Hence, the key step of the determination of the inner product matrix reduces to the root-finding problem in the complex plane with guaranteed error terms solved for instance by the Gourdon-Schönhage algorithm [Gou96]. The full outline of the corresponding Oracle $\mathcal{O}_\gamma$ is given in Algorithm 6.

Theorem 6.1. *Let $\mathbf{K} \cong \mathbf{Q}[X]/(F)$ a number field defined by the polynomial F of height $H(F)$. Fix an order $\mathfrak{o} \in \mathfrak{o}_\mathbf{K}$ and one of its basis γ, given as a list of polynomials of height bounded by $H(B)$. For an ideal $\mathfrak{a}$ given by a $\mathbf{Z}$-basis B in γ,* *the* ADAPTIVE-LLL *operates in at most*

$$O\big[d^3(\log\|B\|_{max} + n_0 + \log(\log H(B) + d\log H(F))) \\ (n_0 + d + \log\|B\|_{max} + \log(\log H(B) + \log H(F)))\mathcal{M}(d)\big],$$

arithmetic operations, with n_0 being the accuracy needed on the representation of the ideal lattice $(\mathfrak{a}, \langle\cdot,\cdot\rangle_\sigma)$to be proper.

Proof. Like in Algorithm 5, the whole complexity is dominated by the last iteration of the outer loop, by superlinear complexity of the arithmetic with regards to the precision used. Let then focus on the body of this loop.

Let first evaluate the (maximal) precision needed on the computation of the roots. With the notations of the theorem, we claim that the entrees of the representation S are bounded by $d^3H(B)^2H(F)^{2(d-1)}$. Indeed, let take $p, q \in \gamma$, represented in $\mathbf{K}$ as polynomials P, Q. Then:

$$\begin{aligned} |\langle p,q\rangle| &\leq \sum_\sigma |\sigma(p)||\sigma(q)| && \text{(Triangle inequality)} \\ &= \sum_{\alpha \text{ root of } P} |P(\alpha)||Q(\alpha)| && \text{(Definition)} \\ &\leq \sum_{\alpha \text{ root of } F} \left(dH(B)\max(|\alpha|,1)^{d-1}\right)^2 && \text{(Triangle inequality)} \\ &\leq \sum_{\alpha \text{ root of } F} \left(dH(B)H(F)^{d-1}\right)^2 && \text{(Cauchy's bound)} \\ &\leq d^3H(B)^2H(F)^{2d-2}. \end{aligned}$$

Hence by denoting as before by n_0 the accuracy needed on the representation of the lattice to be proper, one needs at most $\mathrm{O}(n_0 + \log H(B) + d\log H(F))$ bits of precision on the result of the evaluation of $\langle p,q\rangle$. Since the computation (in floating-point arithmetic) of the polynomial evaluation of one of the $P(\alpha_i)$ or $Q(\alpha_i)$ requires $\Theta(d)$ additions and multiplications, and that the sum over the roots consists on $\Theta(d)$ additions and multiplications, we need at least a $\mathrm{O}(n_0 + \log H(B) + d\log H(F) + d) = \mathrm{O}(n_0 + \log H(B) + d\log H(F))$ bits of precision on the roots $\alpha_1, \dots, \alpha_n$, to ensure a correct computation.

Let then evaluate the complexity of the root-finding sub-procedure of Algorithm 6. The complexity of the root-finding procedure at precision μ for a polynomial of height bounded by H and roots of magnitude at most 2^m is a

$$\mathrm{O}\big(d\log^5 d\log(H+\mu)M(d^2\max(m,d))\big)$$

by the analysis performed in [NR96], therefore, in our case this computation costs at most a

$$\mathrm{O}\big(d\log^5 d\log(H(F) + d\log H(F) + \log H(B) + n_0)M\big(d^2\max(\log(H(F),d))\big)\big),$$

which is negligible with regards to the cost of ADAPTIVE-LLL of:

$$O\big[d^3(\log\|B\|_{\max} + n_0 + \log(\log H(B) + d\log H(F))) \\ (n_0 + d + \log\|B\|_{\max} + \log(\log H(B) + \log H(F)))\mathcal{M}(d)\big].$$

□

6.4. **CM-fields and integrality of the inner product.** A number field $\mathbf{K}$ is a CM-field if it is a quadratic extension ${}^{\mathbf{K}}\!/_{\mathbf{F}}$ where the base field $\mathbf{F}$ is totally real but $\mathbf{K}$ is totally imaginary. Let us present a simple observation on the behavior of the Archimedean embedding in this class of fields.

Lemma 6.1. *Let* $\mathbf{K}$ *a CM-field. Then for any integers* $a, b \in \mathfrak{o}_{\mathbf{K}}$*,*

$$\langle a, b\rangle \in \mathbf{Z}.$$

Proof. Let a, b two integers of $\mathbf{K}$. In this case $\langle a, b\rangle = \operatorname{tr}(a\kappa(b))$, where κ is the complex conjugation of $\mathbf{K}$. Since the complex conjugation acts as a Galois involution over $\mathbf{K}$, and hence $\mathfrak{o}_{\mathbf{K}}$, $\kappa(b)$ is an integer. Whence $\langle a, b\rangle$ is the trace of an integral element and is thus a rational integer itself. □

Therefore when working with a CM-field $\mathbf{K}$, the matrix $\mathcal{G}_\gamma$ of the inner product in a fixed basis of an order, is in fact integral. As a subfield of a CM-field, any totally real field shares also this property. The reduction of such orders can then be performed directly with this matrix and not an approximation, as presented in the second point of Theorem 5.2.

Conversely, when $\mathbf{K}$ is not a subfield of a CM-field, the corresponding matrix is a priori, not integer-valued. This whole framework of Interval Arithmetic approximation is then important to correctly reduce. Up-to-our knowledge, no satisfactory estimation of the precision required on the inner product to reduce ideals appears in the literature. Some authors like Cohen [Coh93] or Belabas [Bel04] suggest using some arbitrary approximation and let the LLL reduction operate. However in this setting, the outputted basis has no reason to be LLL-reduced. In some cases this aspect is not an issue since the reduction was only used to shrink the size of coefficients involved in some computations, but one can't assert any bounds on the norm of elements appearing in these *somewhat reduced*-bases.

6.5. **Theoretical bounds on the precision of the representation.** As already mentioned, the precision needed for the computation of the inner product matrix is difficult to evaluate. Using Lemma 4.1 one can derive some upper bounds depending on the defining polynomial of the number field.

We start in the whole generality by giving a lower bound on the least eigenvalue of a real matrix using its trace and its determinant:

Lemma 6.2 (Bound on the least eigenvalue)**.** *Let* $A \in \mathbf{C}^{d\times d}$ *a complex valued square matrix, with real positive eigenvalues* $0 < \lambda_d \leq \cdots \leq \lambda_1$*, then:*

$$\left[\frac{d-1}{\operatorname{tr} A}\right]^{d-1} \det A \leq \lambda_d \leq (\det A)^{\frac{1}{d}}$$

Proof.
Upper bound: Trivial by the positivity of eigenvalues. Notice that bounding by $\frac{\operatorname{tr} A}{d}$ is also valid by the same argument but slightly less effective by arithmetic-geometric means inequality.

Lower bound: By arithmetic-geometric means inequality and positivity of the eigenvalues we have:

$$\left(\frac{\det A}{\lambda_d}\right)^{\frac{1}{d-1}} \leq \frac{\operatorname{tr} A - \lambda_d}{d-1} \leq \frac{\operatorname{tr} A}{d-1}$$

As such, $\left[\frac{d-1}{\operatorname{tr} A}\right]^{d-1}\frac{\det A}{\lambda_d} \leq 1$. □

Let now d, H two fixed positive integers. Let $P(X) = \sum_{i=0}^{d} a_i X^i \in \mathbf{Z}[X]$ a monic integral polynomial of geometric height bounded by H, that is such that $\max_i |a_{d-i}|^{\frac{1}{i}} \leq H$. Denote by Δ_P its discriminant. Assuming that P is irreducible, consider the set of its complex roots $\theta_1, \ldots, \theta_n$ and construct the associated $d \times d$ Vandermonde matrix $V = V(\theta_1, \ldots, \theta_d)$. Then we have:

Lemma 6.3. *Let A be the Hermitian matrix square $V \cdot V^\dagger$, with strictly positive least eigenvalue $\lambda_d(A) > 0$, then:*

$$\left[\frac{d-1}{d}\right]^{d-1} d\Delta_P^2 \frac{(H^2-1)^{d-1}}{H^{2d^2-2}} \leq \lambda_d(A) \leq \Delta_P^{\frac{1}{d}}$$

Proof.
Upper bound: Recall that by definition of the Vandermonde determinant,

$$\det A = \left[\prod_{1\leq i,j} (\theta_i - \theta_j)\right]^2 = \Delta_P^2.$$

Lower bound: Using Fujiwara's bound—as stated in [Mar66]— we can assert that every root of P has magnitude bounded by H. Using a finite geometric summation then yields:

$$\operatorname{tr} A = \sum_{1\leq i,j\leq d} |\theta_i|^{2j} \leq d\frac{H^{2d+2}-H}{H^2-1}.$$

□

By Proposition 4.1 we derive:

Corollary 6.1. *The accuracy of the minimal proper representation of the Gram-matrix representing a principal order of the number field $\mathbf{Q}[X]/P(X)$ in its power basis satisfies*

$$n_0 \leq \log d - 2\log \Delta_P + 2(d^2 - d + o(d))\log H$$

The previous bound is exponential and gives little information on the typical behavior of the accuracy of a random polynomial or number field. Experiments seem to show that the accuracy needed is nonetheless small with regards to the degree: only a logarithmic number of bits appear necessary to ensure the properness of the representation. The following heuristic encompasses the numerous experiments conducted.

Heuristic 6.1. *Let $P \in \mathbb{Z}[X]$ a monic integral polynomial of height $H > 1$ and degree d, taken uniformly at random that is each of the d free coefficients of P is sampled uniformly and independently in the set $\{-H, \ldots, +H\}$. Construct V as before, then the expected precision of the minimal representation of the Gram-matrix $V \cdot V^\dagger$ under this distribution is*

$$\mathbb{E}_P\left[n_0(V \cdot V^\dagger)\right] = \theta(\log n).$$

6.6. **Relation with the Expansion Factor.** Let $\mathbf{K}$ be a number field of degree d and $\mathfrak{o}$ an order of $\mathbf{K}$. Let us fix a integral basis $\gamma = (b_1, \dots, b_d)$ of $\mathfrak{o}$. Define the coefficient embedding of $\mathfrak{o}$ relatively to γ as the injective morphism:

$$\iota_\gamma : \left| \begin{array}{ccc} \mathfrak{o} & \longrightarrow & \mathbf{Z}^n \\ a = \sum_{i=1}^d a_i b_i & \longmapsto & (a_1, \cdots, a_d)^T \end{array} \right.$$

that is the map taking the decomposition of an element of the order in the basis γ and mapping it to the vector of corresponding coefficients. Taking the infinity norm through the embedding ι_γ makes arise the so-called (coefficient) infinity norm $\|\cdot\|_{\gamma,\infty}$, formally defined as:

$$\|a\|_{\gamma,\infty} = \|\iota_\gamma(a)\|_{\max}.$$

Definition 6.1 (Expansion Factor)**.** *Let* $\mathbf{K}$ *be a number field and* $\mathfrak{o}$ *an order of* $\mathbf{K}$*. The expansion factor of* $\mathfrak{o}$ *in basis* $\gamma \subset \mathfrak{o}$ *is defined as:*

$$\delta_{\mathfrak{o},\gamma} = \sup_{a,b \in \mathfrak{o}} \frac{\|ab\|_{\gamma,\infty}}{\|a\|_{\gamma,\infty}\|b\|_{\gamma,\infty}}$$

Intuitively the expansion factor gives a measurement of the obstruction— induced by the ring structure of $\mathfrak{o}$—on the infinity norm to be sub-multiplicative, that is satisfying $\|ab\| = \|a\|\|b\|$ for any a, b. The following lemma relates the canonical embedding to the expression factor through the eigenvalues of the Gram-matrix introduced in Section 6.3.

Proposition 6.1. *Let* $\mathbf{K}$ *be a number field and* $\mathfrak{o}$ *an order of* $\mathbf{K}$ *of basis* γ*. Denote by* $\mathcal{G}_\gamma$ *the Gram-matrix* $\mathcal{G}_\gamma$ *of* γ *and by* λ_1 *(resp.* λ_d*) its greatest (resp. least) eigenvalue. Then:*

$$\delta_{\mathfrak{o},\gamma} \leq \frac{\lambda_1}{\sqrt{\lambda_d}} \deg \mathbf{K}$$

Proof. Recall above else all that $\|\cdot\|$ denotes the norm induced by the canonical embedding. Remark that this norm is an algebra-norm:

$$\begin{aligned} \|ab\| &= \sum_\sigma \sigma(ab)\overline{\sigma(ab)} \\ &= \sum_\sigma \sigma(a)\sigma(b)\overline{\sigma(a)\sigma(b)} \\ &\leq \left(\sum_\sigma \sigma(a)\overline{\sigma(a)}\right)\left(\sum_\sigma \sigma(b)\overline{\sigma(b)}\right) = \|a\|\|b\| \end{aligned}$$

Let now $a, b \in \mathfrak{o}$ and $A = \iota_\gamma(a), B = \iota_\gamma(b)$ their coefficient embedding. Then, using the characterization of the least (resp. the greatest) eigenvalue of an Hermitian matrix S as the infimum (resp. supremum) over complex vectors of the Rayleigh quotient $\frac{X^T \mathcal{G}_\gamma X}{X^T X}$ yields the norm equivalence:

$$\lambda_d \|a\|^2_{\gamma,\infty} = \lambda_d A^T A \leq \|a\|^2 \leq \lambda_1 A^T A = d\lambda_1 \|a\|^2_{\gamma,\infty}$$

Whence, plugging the sub-multiplicativity proved earlier directly on the product ab yields:

$$\lambda_d \|ab\|^2_{\gamma,\infty} \leq \|ab\|^2 \leq \|a\|^2\|b\|^2 \leq d^2\lambda_1^2 \|a\|^2_{\gamma,\infty}\|b\|^2_{\gamma,\infty}$$

$\square$

Informally we have seen through this paper that the hardness of performing lattice reduction in an order $\mathfrak{o}$ is related to the *ill-conditioning* of the Gram-matrix representing the inner product of the lattice: the smallest the least eigenvalue is, the greater the accuracy must be to represent the lattice and then the slower goes the reduction. Similarly, Proposition 6.1 states that the ill-conditioning of the matrix makes cryptographic constructions (in the sense of [Gen09] for instance) harder to construct. These two—apparently unrelated—problems are actually two avatars of the problem of finding convenient basis of orders to design fast algorithms from reduction theory on the one hand and from arithmetic considerations on the other.

6.7. **The Cyclotomic case.** Cyclotomic fields are of the utmost importance in algebraic number theory as they form a class of fields with nice and important properties. They are indeed related to many problems—Fermat's last theorem, primes in arithmetic progressions to name but a few—but also being the cornerstone of the abelian Galois theory by the Kronecker–Weber theorem (see [Was97] for a more exhaustive reference on the applications of Cyclotomic fields to number theory).

Furthermore, cyclotomic fields have lately become ubiquitous in the cryptographical setting since lattice-based schemes are almost always instantiated in these fields. Indeed due to the existence of a Fast Fourier Transform, the arithmetic in their ring of integers enjoys fast algorithms. Reduction theory in this context is then a practical mean for estimating security parameters of these cryptosystems.

Let $\mathbf{Q}(\zeta_d)$ be the d-cyclotomic field, where classically ζ_d denotes a primitive d-roots of unity. Its degree is then $\varphi(d)$ and its maximal order is precisely the equation order $\mathbf{Z}[\zeta_d]$ (see [Was97] for a proof of this statement). As before, denote by V_σ the Vandermonde matrix $V\left((\zeta_d^{(j)})_{j\in\mathbf{Z}_d^*}\right)$ of primitive roots of unity. Remark that the matrix $V \cdot V^\dagger$ is circulant of first row

$$C(n) = [c_n(0), \ldots, c_n(\varphi(n)-1)]$$

where

$$c_k(n) = \sum_{l\in\mathbb{Z}_n^*}^{n} \zeta_n^{kl}$$

is the *k-th Ramanujan's sum* associated to d. Since $V \cdot V^\dagger$ is circulant of order $\varphi(n)$, classically its eigenvalues are the coefficients of the Discrete Fourier Transform of the polynomial $P(X) = \sum_{i=0}^{\varphi(n)} c_n(i)X^i$, that is the evaluations of P over the $\varphi(n)$-th roots of unity. Since a cyclotomic field is above all a CM-field, the coefficients of V_σ are integers, asserting the integrality of Ramanujan's sums.

6.7.1. *Elementary properties of the sums.* Nonetheless by a finer analysis of these sums, one can derive closed expressions using elementary arithmetic functions, namely Euler's totient[12] φ and Möbius's function[13] μ.

[12]This function evaluated on an integer n counts the positive integers up to n that are relatively prime to n.

[13]The Möbius function is defined as the sum of the primitive n-th roots of unity.

Lemma 6.4 (Prime power case)**.** *Let p be an odd prime and and $1 \leq \ell < p$ an integer. Then we have:*

$$c_{p^\ell}(k) = \begin{cases} \varphi(p^\ell)\big(= p^{\ell-1}(p-1)\big), & \text{if } k = 0 \mod n \\ -p^{\ell-1}, & \text{if } k = p^i \mod n \text{ with } i \neq \ell \\ 0, & \text{otherwise} \end{cases}$$

Lemma 6.5 (Multiplicativity)**.** *Let a, b two coprime non-negative integers then*

$$\forall k \in \mathbf{N}, \quad c_{ab}(k) = c_a(k)c_b(k)$$

Lemma 6.6. *For any non-negative integer n, we have:*

$$\forall k \in \mathbf{N}, c_n(k) = \mu\left(\frac{n}{gcd(k,n)}\right)\frac{\varphi(n)}{\varphi\left(\frac{n}{gcd(k,n)}\right)}$$

Proof. By multiplicativity and Lemma 6.4. □

Lemma 6.7 (Power-of-two and Prime Cyclotomics)**.**

(1) Let $d > 1$ an integer. The least eigenvalue of the Gram-matrix of the order $\mathbf{Z}[\zeta_{2^d}]$ is $\varphi(n)$.

(2) Let p a prime. The least eigenvalue of the Gram-matrix of the order $\mathbf{Z}[\zeta_p]$ is 1

Proof.

(1) Using the characterization of Lemma 6.4, the vector $C(2^d)$ has all but its first coordinate equal to zero. Since its first coordinate is $\varphi(2^d) = 2^{d-1}$ the spectrum of the matrix $VV^\dagger$ is reduced to $\{2^{d-1}\}$.

(2) Using the characterization of Lemma 6.4, the vector $C(p)$ has all but its first coordinate equal to -1. The matrix $VV^\dagger$ then decomposes as $p\mathbf{Id} - \mathbf{1}$ where $\mathbf{1}$ is the matrix of size $p-1 \times p-1$ of all coefficients equal to one. Since the latter matrix verifies $\mathbf{1}^2 = (p-1)\mathbf{1}$, we can deduce that its spectrum is $\{0, p-1\}$, the zero eigenvalue being repeated $p-2$ times. Hence, since $p\mathbf{Id}$ and $\mathbf{1}$ trivially commute, we conclude that the spectrum of $V^\dagger V$ is $\{1, p-1\}$.

□

Theorem 6.2. *Let $\mathbf{K}$ the d-cyclotomic field and $\mathfrak{o}_\mathbf{K}$ its ring of integer. Let $(\mathfrak{a}, \langle\cdot,\cdot\rangle)$ a rank d lattice arising from an ideal $\mathfrak{a}$ of $\mathfrak{o}_\mathbf{K}$, exactly represented by the pair (G, A) in the power-basis $\zeta \in \Lambda^{\varphi(d)}$. The* ADAPTIVE-LLL *(Algorithm 5) operates in at most*

$$O\big(\varphi(d)^3(\log\|B\|_{max} + \log\varphi(d))(\varphi(d) + \log\|B\|_{max} + \log\varphi(d))\mathcal{M}(d)\big)$$

arithmetic operations.

If $d = 2^n$ is a power of two, the above complexity collapses to:

$$O\big(2^{3n}\log\|B\|_{max}(2^n + \log\|B\|_{max})\mathcal{M}(d)\big)$$

arithmetic operations.

Proof. The first point is a direct application of Theorem 5.2 together with the bound: $\|S\|_{\max} = \max_i C(d)_i = \phi(n)$ easily derived from Lemma 6.6. The second follows from the computation of the least eigenvalue in the power-of-two case of Lemma 6.7 and dividing the matrix S by $\phi(n)$. □

Remark that the computation of the least eigenvalue in the prime case in Lemma 6.7 asserts it is not possible to divide the matrix S to get a smaller representation with the technique of Section 4.3.1.

Thomas: Section sur l'implem?

References

[ABC+17] Martin Albrecht, Shi Bai, D. Cadé, Xaver Pujol, and Damien Stehlé. fplll-5.0, a floating-point LLL implementation. Available at `http://perso.ens-lyon.fr/damien.stehle`, 2017.

[B+99] Dan Boneh et al. Twenty years of attacks on the RSA cryptosystem. *Notices of the AMS*, 46(2):203–213, 1999.

[Bel04] Karim Belabas. Topics in computational algebraic number theory. *J. Théor. Nombres Bordeaux*, 16:19–63, 2004.

[BF13] Jean-François Biasse and Claus Fieker. Improved techniques for computing the ideal class group and a system of fundamental units in number fields. *The Open Book Series*, 1(1):113–133, 2013.

[Coh93] Henri Cohen. *A Course in Computational Algebraic Number Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1993.

[EFGK16] Thomas Espitau, Pierre-Alain Fouque, Alexandre Gélin, and Paul Kirchner. Computing generator in cyclotomic integer rings. *IACR Cryptology ePrint Archive*, 2016:957, 2016.

[Elk00] Noam D. Elkies. Rational points near curves and small nonzero $|x^3-y^2|$ via lattice reduction. *Algorithmic Number Theory: 4th International Symposium, ANTS-IV Leiden, The Netherlands, July 2-7, 2000. Proceedings*, pages 33–63, 2000.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009*, pages 169–178, 2009.

[GJ16] Alexandre Gélin and Antoine Joux. Reducing number field defining polynomials: an application to class group computations. In *Algorithmic Number Theory Symposium XII*, volume 19 of *LMS Journal of Computation and Mathematics*, pages 315–331, 2016.

[GN08] Nicolas Gama and Phong Q. Nguyen. Finding short lattice vectors within Mordell's inequality. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 207–216, 2008.

[Gou96] Xavier Gourdon. Combinatoire, algorithmique et géométrie des polynomes. *PhD thesis*, pages 27–49, 1996.

[HJ12] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, New York, NY, USA, 2nd edition, 2012.

[HMM98] George Havas, Bohdan S. Majewski, and Keith R. Matthews. Extended GCD and Hermite normal form algorithms via lattice basis reduction. *Experimental Mathematics*, 7(2):125–136, 1998.

[Jäg05] G. Jäger. Reduction of Smith normal form transformation matrices. *Computing*, 74(4):377–388, 2005.

[JKDW01] Luc Jaulin, Michel Kieffer, Olivier Didrit, and Eric Walter. *Applied interval analysis: with examples in parameter and state estimation, robust control and robotics*. Springer Verlag, 2001.

[Kal83] Erich Kaltofen. On the complexity of finding short vectors in integer lattices. In J. A. van Hulzen, editor, *Computer Algebra, EUROCAL '83, European Computer Algebra Conference*, volume 162 of *Lecture Notes in Computer Science*, pages 236–244. Springer, 1983.

[Len83] Hendrik W. Jr. Lenstra. Integer programming with a fixed number of variables. *Math. Oper. Res.*, 8:538–548, 1983.

[LLL82] Arjen K. Lenstra, Hendrik W. Jr. Lenstra, and Lászlo Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982.

[LO85] J. C. Lagarias and A. M. Odlyzko. Solving low-density subset sum problems. *J. ACM*, 32(1):229–246, 1985.

[LS92] László Lovász and Herbert E. Scarf. The generalized basis reduction algorithm. *Math. Oper. Res.*, 17(3):751–764, 1992.

[LS17] Hendrik W. Jr. Lenstra and Alice Silverberg. Lattices with symmetry. *Journal of Cryptology*, 30(3):760–804, Jul 2017.

[Mar66] M. Marden. *Geometry of polynomials – Second Edition.* Number 3 in Geometry of Polynomials. American Mathematical Society, 1966.

[Moo62] Ramon Moore. *Interval Arithmetic and automatic error analysis in digital computing.* PhD thesis, Stanford, 1962.

[Moo77] Ramon E. Moore. *Methods and applications of interval analysis.* 1977.

[NR96] C.Andrew Neff and John H. Reif. An efficient algorithm for the complex roots problem. *Journal of Complexity*, 12(2):81 – 115, 1996.

[NS06] Phong Q. Nguyen and Damien Stehlé. LLL on the average. In Florian Hess, Sebastian Pauli, and Michael E. Pohst, editors, *Algorithmic Number Theory, 7th International Symposium, ANTS-VII,*, volume 4076 of *Lecture Notes in Computer Science*, pages 238–256. Springer, 2006.

[NS09] Phong. Q. Nguyen and Damien Stehlé. An LLL algorithm with quadratic complexity. *SIAM J. of Computing*, 39(3):874—-903, 2009.

[OR85] A.M. Odlyzko and H.J.J. Riele. Disproof of the Mertens conjecture. *Journal für die reine und angewandte Mathematik*, 357:138–160, 1985.

[RR88] Helmut Ratschek and Jon Rokne. *New Computer Methods for Global Optimization.* Halsted Press, New York, NY, USA, 1988.

[Sch87] Claus-Peter Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theor. Comput. Sci.*, 53:201–224, 1987.

[Sch88] Claus-Peter Schnorr. A More Efficient algorithm for lattice basis reduction. *J. Algorithms*, 9(1):47–62, 1988.

[SE94] Claus-Peter Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Program.*, 66:181–199, 1994.

[Sho17] V. Shoup. NTL 10.5.0, a library for doing number theory. Available at `http://www.shoup.net/ntl/`, 2017.

[Sun09] Teruo Sunaga. Theory of an interval algebra and its application to numerical analysis. *Japan J. Indust. Appl. Math.*, 26, 10 2009.

[Was97] Lawrence C. Washington. *Introduction to Cyclotomic Fields.* Graduate Texts in Mathematics. Springer New York, 1997.

[Weg87] B. M. M. De Weger. Solving exponential Diophantine equations using lattice basis reduction algorithms. *J. Number theory*, 26:325–367, 1987.

[Wey12] Hermann Weyl. Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen (mit einer Anwendung auf die

Theorie der Hohlraumstrahlung). *Mathematische Annalen*, 71(4):441–479, Dec 1912.

[You31] Rosalind Cecil Young. *The algebra of many-values quantities.* PhD thesis, Cambridge, 1931.

Sorbonne Université, lip 6 umr 7606, paris, france
E-mail address: `t.espitau@gmail.com`

Chaire de cryptologie de la fondation de l'upmc; sorbonne université, imj-prg, paris, france
E-mail address: `antoine.joux@m4x.org`