H-LLL: Using Householder Inside LLL

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

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

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

ABSTRACT

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

Categories and Subject Descriptors

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

General Terms

Algorithms

Keywords

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

1. INTRODUCTION

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

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

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

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

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

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

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

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

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

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

2. LATTICE REDUCTION

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

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

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

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

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

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

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

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

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

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

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

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

Algorithm 1 A generic LLL algorithm.

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

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

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

Algorithm 2 Householder's algorithm.

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

- $R := \diamond(B)$.
- For i from 1 to d, do
  - For j from 1 to $i-1$, do
    - $\mathbf{r}_i[j..n] = \mathbf{r}_i[j..n] - (\mathbf{v}_j^T \mathbf{r}_i[j..n]) \cdot \mathbf{v}_j; \mathbf{r}_i[j] := \sigma_j \mathbf{r}_i[j]$.
  - $\mathbf{r} := \mathbf{r}_i[i..n]; \mathbf{v}_i := \mathbf{r}$.
  - $\sigma_i := \operatorname{sign}(\mathbf{r}[1]);\ s := \sigma_i \|\mathbf{r}\|$.
  - $\mathbf{v}_i[1] := (-\sum_{j=2}^{n-i+1} \mathbf{r}[j]^2)/(\mathbf{r}[1] + s)$.
  - If $\mathbf{v}_i[1] \neq 0$, then $\mathbf{v}_i := \mathbf{v}_i / \sqrt{-s \cdot \mathbf{v}_i[1]}$.
  - $\mathbf{r}_i[i..n] := (\|\mathbf{r}\|, 0, \ldots, 0)^T$.
- Return the first d rows of R.

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

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

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

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

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

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

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

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

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

and

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

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

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

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

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

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

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

3. AN INCOMPLETE SIZE-REDUCTION

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

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

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

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

Algorithm 3 The incomplete size-reduction algorithm.

Input: A matrix $B \in \mathbb{Z}^{n \times d}$, $\kappa \leq d$ and the output $\overline{\mathbf{r}}_1, \ldots, \overline{\mathbf{r}}_{\kappa-1}, \overline{\mathbf{v}}_1, \ldots, \overline{\mathbf{v}}_{\kappa-1}, \sigma_1, \ldots, \sigma_{\kappa-1}$ of Algorithm 2 when given as input the first $\kappa - 1$ columns of B. We assume that the first $\kappa - 1$ columns of B are (δ, η, θ)-LLL-reduced with $\eta \in (1/2, 1)$, $\delta \in (\eta^2, 1)$ and $\theta \in (0, \eta - 1/2)$.
Input: $\diamond(2^{-cd})$ (for an arbitrary $c > 0$) and a fp precision $p > \log_2(2^{\frac{cd}{2}+9}\kappa^3\phi(\kappa)\alpha/\theta)$.
1: Do
2: Compute $\overline{\mathbf{r}}_\kappa$ using Steps 3–4 of Algorithm 2.
3: For i from $\kappa - 1$ to 1, do
4: $X_i = \left\lfloor \diamond\left(\frac{\overline{R}_{i,\kappa}}{\overline{R}_{i,i}}\right)\right\rceil$.
5: For j from 1 to $i-1$, do $\overline{R}_{j,\kappa} := \diamond\left(\overline{R}_{j,\kappa} - \diamond\left(X_i \overline{R}_{j,i}\right)\right)$.
6: $t := \diamond(\|\mathbf{b}_\kappa\|^2)$; $\mathbf{b}_\kappa := \mathbf{b}_\kappa - \sum_{i<\kappa} X_i \mathbf{b}_i$.
7: Until $\diamond(\|\mathbf{b}_\kappa\|^2) > \diamond(\diamond(2^{-cd}) \cdot t)$.
8: Compute $\overline{\mathbf{r}}_\kappa, \overline{\mathbf{v}}_\kappa, \sigma_\kappa$ using Steps 3–9 of Algorithm 2.
9: Return B, $\overline{\mathbf{r}}_1, \ldots, \overline{\mathbf{r}}_\kappa$, $\overline{\mathbf{v}}_1, \ldots, \overline{\mathbf{v}}_\kappa$ and $\sigma_1, \ldots, \sigma_\kappa$.

3.1 Analysis of Steps 3–6

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

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

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

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

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

Therefore, by using Corollary 2.4:

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

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

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

which completes the proof. ☐

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

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

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

columns of B, we have:

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

The result is then provided by Lemma 3.1. □

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

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

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

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

which provides the result. □

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

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

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

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

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

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

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

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

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

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

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

This completes the proof. □

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

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

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

Proof. Using Lemma 3.4, we have:

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

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

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

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

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

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

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

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

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

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

which completes the proof. □

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

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

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

Proof. Using Lemma 3.6:

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

The latter provides the result. □

3.2 Correctness and Cost of Algorithm 3

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

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

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

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

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

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

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

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

For the second inequality, note that Lemma 3.6 implies:

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

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

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

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

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

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

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

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

Furthermore, using Lemma 3.6, we get (noting $\mathbf{a} = (R^e_{1,\kappa},\ldots,R^e_{i_0,\kappa},0,\ldots,0)$ and $\mathbf{b} = (0,\ldots,0,R^e_{i_0+1,\kappa},\ldots,R^e_{\kappa,\kappa},0,\ldots,0)$):

$$\begin{aligned}\|\mathbf{b}^e_\kappa\| - \|\mathbf{b}^\ell_\kappa\| &= \|\mathbf{r}^e_\kappa\| - \|\mathbf{r}^\ell_\kappa\| \\ &\leq \|\mathbf{a}\| + \|\mathbf{b}\| - \|\mathbf{b}\| \\ &\leq \sum_{i\leq i_0}|R^e_{i,\kappa}| \\ &\leq (\kappa\alpha^{i_0} + \overline{\theta})R_{i_0,i_0} + \overline{\theta}\|\mathbf{b}^\ell_\kappa\| \\ &\leq 9(\kappa\alpha^{i_0} + \overline{\theta})\|\mathbf{b}^\ell_\kappa\|.\end{aligned}$$

This gives that $\|\mathbf{b}^e_\kappa\| \leq 10\kappa\alpha^\kappa\|\mathbf{b}^\ell_\kappa\|$, which provides the bound. □

The result above leads us to the following complexity upper bound.

THEOREM 3.11. *Let* $(\mathbf{b}_1,\ldots,\mathbf{b}_d) \in \mathbb{Z}^{n\times d}$ *be a valid input to Algorithm 3. Let* κ *be the input index. Suppose the precision satisfies* $p > \log_2(2^{\frac{cd}{2}+9}\kappa^3\phi(\kappa)\alpha/\theta)$ *and* $p = 2^{O(d)}$*. Then the execution finishes within*

$$O\left[\left(d + \log\frac{\|\mathbf{b}^b_\kappa\|}{\|\mathbf{b}^e_\kappa\|}\right)\frac{n\mathcal{M}(d)}{d}(d + \log\|B\|)\right] \text{ bit operations,}$$

where $\|B\| = \max_{i\leq\kappa}\|\mathbf{b}_i\|$ *and* $\mathbf{b}^b_\kappa$ *(resp.* $\mathbf{b}^e_\kappa$*) is* $\mathbf{b}_\kappa$ *at the beginning of Step 1 (resp. Step 9).*

Proof. The bit-cost of one iteration of Steps 4 and 5 is $O(d\mathcal{M}(d))$ for handling the mantissas (thanks to the second restriction on p) and $O(d\log(d+\log\|B\|))$ for handling the exponents (thanks to Corollary 2.4 and Lemmata 3.1 and 3.4). This implies that one iteration of the loop made of Steps 3–5 costs $O(d^2\mathcal{M}(d)+d^2\log\log\|B\|)$. A similar bound $O(nd\mathcal{M}(d)+nd\log\log\|B\|)$ holds for one iteration of Step 2. The computation of t at Step 6 is negligible compared to the costs above. Theorem 3.9 implies that the update of $\mathbf{b}_\kappa$ at Step 6 can be performed within $O(n\mathcal{M}(d)\log(d\|B\|))$ bit operations (note that though X_i can be a very large integer, it is stored on $\leq p = O(d)$ bits). The cost of Step 7 is also negligible compared to the costs above. Overall, the bit-cost of one iteration of the loop consisting of Steps 1–7 is $O(n\mathcal{M}(d)(d+\log\|B\|))$. Lemma 3.10 provides the result. □

4. AN LLL RELYING ON HOUSEHOLDER'S ALGORITHM

The H-LLL algorithm (Algorithm 4) follows the general structure of LLL algorithms (see Algorithm 1). For the size-reduction, it relies on Algorithm 3. The precision requirement is a little stronger than in the previous section. Asymptotically, for close to optimal parameters δ, η and θ (i.e., $\delta \approx 1$, $\eta \approx 1/2$ and $\theta \approx 0$), a sufficient precision is $p \approx d$.

Algorithm 4 The H-LLL algorithm.

Input: A matrix $B \in \mathbb{Z}^{n\times d}$ of rank d and valid LLL parameters δ, η and θ, with $\theta < \eta - 1/2$.
Input: $\diamond(2^{-cd})$ (for an arbitrary $c > 0$) and a fp precision $p > p_0 + 1 - \log_2(1-\delta) - \log_2(\eta - \theta - 1/2)$ with $p_0 := \log_2(d^3\phi(d)\alpha^d/\theta) + 16 + cd/2$.
Output: A (δ,η,θ)-LLL-reduced basis of the lattice spanned by the columns of B.

1: Let $\overline{\delta}$ be a fp number in $(\delta + 2^{-p+p_0}, 1 - 2^{-p+p_0})$.
2: Compute $\overline{\mathbf{r}}_1, \overline{\mathbf{v}}_1, \sigma_1$ using Steps 3–9 of Algorithm 2.
3: $\kappa := 2$. While $\kappa \le d$, do
4: Call Algorithm 3 on $B, \overline{\mathbf{r}}_1, \ldots, \overline{\mathbf{r}}_{\kappa-1}, \overline{\mathbf{v}}_1, \ldots, \overline{\mathbf{v}}_{\kappa-1}$ and $\sigma_1, \ldots, \sigma_{\kappa-1}$.
5: $s := \diamond(\|\diamond(\mathbf{b}_\kappa)\|^2)$; $s := \diamond(s - \sum_{i\le\kappa-2}\overline{R}_{i,\kappa}^2)$.
6: If $\diamond(\overline{\delta}\cdot\diamond(\overline{R}_{\kappa-1,\kappa-1}^2)) \le s$, then $\kappa := \kappa+1$.
7: Else swap $\mathbf{b}_{\kappa-1}$ and $\mathbf{b}_\kappa$; $\kappa := \max(\kappa-1, 2)$.
8: Return B.

Before proceeding to the analysis of Algorithm 4, let us explain how Step 5 is performed. We compute $\diamond(\|\diamond(\mathbf{b}_\kappa)\|^2)$ sequentially; we compute the $\diamond(\overline{R}_{i,\kappa}^2)$'s; and finally we compute $s := \diamond(\|\diamond(\mathbf{b}_\kappa)\|^2 - \sum_{i\le\kappa-2}\overline{R}_{i,\kappa}^2)$ sequentially. Corollary 2.4 and Theorem 3.9 provide the soundness of such a computation.

LEMMA 4.1. *Assume that the first $\kappa-1$ columns of B are LLL-reduced. Then at the end of Step 5, we have:*

$$\left|s - (R_{\kappa,\kappa}^2 + R_{\kappa-1,\kappa}^2)\right| \le 2^{-p+12}\kappa^3\alpha^\kappa\phi(\kappa)(R_{\kappa,\kappa}^2 + R_{\kappa-1,\kappa-1}^2).$$

Proof. First of all, thanks to [4, Eq. (3.5)], we have $|\diamond\|\diamond(\mathbf{b}_\kappa)\|^2 - \|\mathbf{b}_\kappa\|^2| \le n2^{-p+1}\|\mathbf{b}_\kappa\|^2$. Also:

$$\begin{aligned}|\diamond(\overline{R}_{i,\kappa}^2) - R_{i,\kappa}^2| &\le 2^{-p+1}R_{i,\kappa}^2 + 2|\overline{R}_{i,\kappa}^2 - R_{i,\kappa}^2| \\ &\le 2^{-p+1}R_{i,\kappa}^2 + 2\Delta R_{i,\kappa}(|R_{i,\kappa}| + \Delta R_{i,\kappa}).\end{aligned}$$

Thanks to the LLL-reducedness of the first $\kappa-1$ columns of B, Corollary 2.4 and Theorem 3.9, we have (using $\overline{\theta} \le \alpha^{-\kappa}$):

$$\begin{aligned}|R_{i,\kappa}| &\le 2(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + \alpha^{-\kappa}\|\mathbf{b}_\kappa\|) \\ &\le 8\kappa(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + R_{\kappa,\kappa}) \\ \Delta R_{i,\kappa} &\le 2^{-p}\phi(i)(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + \|\mathbf{b}_\kappa\|) \\ &\le 2^{-p+2}\kappa\phi(i)(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + R_{\kappa,\kappa}).\end{aligned}$$

As a consequence, we obtain the bound:

$$\begin{aligned}|\diamond(\overline{R}_{i,\kappa}^2) - R_{i,\kappa}^2| &\le 2^{-p+8}\kappa^2\alpha^{2\kappa}(R_{\kappa-1,\kappa-1}^2 + R_{\kappa,\kappa}^2) \\ &+2^{-p+7}\kappa^2\phi(i)(\alpha^{\kappa-i}R_{\kappa-1,\kappa-1} + R_{\kappa,\kappa})^2 \\ &\le 2^{-p+9}\kappa^2\alpha^\kappa\phi(\kappa)(R_{\kappa-1,\kappa-1}^2 + R_{\kappa,\kappa}^2).\end{aligned}$$

Finally, using [4, Eq. (3.5)], we get the bound:

$$\begin{aligned}|s - (R_{\kappa,\kappa}^2 + R_{\kappa-1,\kappa-1}^2)| &\le \kappa 2^{-p+1}(R_{\kappa,\kappa}^2 + R_{\kappa-1,\kappa-1}^2) \\ &+ 2|\diamond\|\mathbf{b}_\kappa\|^2 - \|\mathbf{b}_\kappa\|^2| + 2\sum_{i\le\kappa-2}|\diamond(\overline{R}_{i,\kappa}^2) - R_{i,\kappa}^2|,\end{aligned}$$

which leads to the result. □

LEMMA 4.2. *Assume that the first $\kappa-1$ columns of B are LLL-reduced. Then at the end of Step 5, we have:*

$$|\diamond(\bar{\delta}\cdot\diamond(\bar{R}_{\kappa-1,\kappa-1}^2)) - \bar{\delta}R_{\kappa-1,\kappa-1}^2| \le 2^{-p+3}\phi(\kappa)\bar{\delta}R_{\kappa-1,\kappa-1}^2.$$

Lemmata 4.1 and 4.2 imply the soundness of the test of Step 6.

THEOREM 4.3. *Let $\overline{\theta} = 2^{-p+8+\frac{cd}{2}}d^3\phi(d)$ and $\overline{\eta} = 1/2 + 2^{-p+1}\phi(d)$. Assume that the first $\kappa-1$ columns of B are (δ,η,θ)-LLL-reduced. If the test of Step 6 succeeds then the first κ columns of B are (δ,η,θ)-LLL-reduced. Otherwise $\delta' R_{\kappa-1,\kappa-1}^2 > R_{\kappa,\kappa}^2 + R_{\kappa-1,\kappa}^2$ with $\delta' = \overline{\delta}(1 + 2^{-p+14}\kappa^3\phi(\kappa)\alpha^\kappa)$.*

Proof. Suppose that the test succeeds. Corollary 2.4 and Lemmata 4.1 and 4.2 imply:

$$\begin{aligned}&(1 - 2^{-p+3}\phi(\kappa))\overline{\delta}R_{\kappa-1,\kappa-1}^2 \\ &\le (1 + 2^{-p+12}\kappa^3\alpha^\kappa\phi(\kappa))(R_{\kappa,\kappa}^2 + R_{\kappa-1,\kappa-1}^2).\end{aligned}$$

By choice of $\overline{\delta}$, this implies that $\delta R_{\kappa-1,\kappa-1}^2 \le R_{\kappa-1,\kappa}^2 + R_{\kappa,\kappa}^2$.
Now, using Theorem 3.9, we know that:

$$\begin{aligned}|R_{\kappa-1,\kappa}| &\le (\overline{\eta} + \overline{\theta})R_{\kappa-1,\kappa-1} + \overline{\theta}\|\mathbf{b}_\kappa\| \\ &\le (\overline{\eta} + \overline{\theta}(1 + 3\kappa\alpha^\kappa))R_{\kappa-1,\kappa-1} + 3\overline{\theta}\kappa R_{\kappa,\kappa} \\ &\le \eta R_{\kappa-1,\kappa-1} + \theta R_{\kappa,\kappa}.\end{aligned}$$

As a consequence, we have $R_{\kappa-1,\kappa-1} \le \alpha R_{\kappa,\kappa}$. By using Theorem 3.9 again, we have:

$$\begin{aligned}|R_{i,\kappa}| &\le \overline{\eta}R_{i,i} + \overline{\theta}(\|\mathbf{b}_\kappa\| + R_{\kappa-1,\kappa-1}) \\ &\le \overline{\eta}R_{i,i} + \overline{\theta}(3\kappa\max_{j\le\kappa}(R_{j,j}) + \alpha R_{\kappa,\kappa}) \\ &\le \overline{\eta}R_{i,i} + 4\overline{\theta}\kappa\alpha^\kappa R_{\kappa,\kappa},\end{aligned}$$

which completes the proof of the first claim of the theorem.
Suppose now that the test fails. Corollary 2.4 and Lemmata 4.1 and 4.2 imply:

$$\begin{aligned}&(1 + 2^{-p+3}\phi(\kappa))\overline{\delta}R_{\kappa-1,\kappa-1}^2 \\ &\ge (1 - 2^{-p+12}\kappa^3\alpha^\kappa\phi(\kappa))(R_{\kappa,\kappa}^2 + R_{\kappa-1,\kappa-1}^2).\end{aligned}$$

By definition of δ', this implies that $\delta' R_{\kappa-1,\kappa-1}^2 > R_{\kappa-1,\kappa}^2 + R_{\kappa,\kappa}^2$. □

We can now conclude our study of Algorithm 4.

THEOREM 4.4. *Algorithm 4 returns a (δ,η,θ)-LLL-reduced basis $(\mathbf{b}_1^e, \ldots, \mathbf{b}_d^e)$ of the lattice spanned by the input basis $(\mathbf{b}_1^b, \ldots, \mathbf{b}_d^b) \in \mathbb{Z}^{n\times d}$. Furthermore, the bit complexity is*

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

where $\|B\| = \max\|\mathbf{b}_i\|$ and d_i^b (resp. d_i^e) is the determinant of the lattice spanned by the first i columns of the input (resp. output) basis. The complexity bound above is itself $O(nd^2\mathcal{M}(d)\log\|B\|(d + \log\|B\|))$.

Proof. Using the classical analysis of the LLL algorithm [7] and Theorem 4.3, we know that the algorithm terminates within $O\left(d + \log \prod_{i \leq d} \frac{d_i^b}{d_i^e}\right)$ iterations. A simple induction using Theorem 4.3 proves that the output is indeed (δ, η, θ)-LLL reduced. Furthermore, the classical analysis of LLL yields that at any moment, the norms of the basis vectors are below $d\|B\|$ (except within the calls to Algorithm 3). Each call to Algorithm 3 that transforms $\mathbf{b}_\kappa^{(old)}$ into $\mathbf{b}_\kappa^{(new)}$ costs

$$O\left[\left(d + \log \frac{\|\mathbf{b}_\kappa^{(old)}\|}{\|\mathbf{b}_\kappa^{(new)}\|}\right) \frac{n\mathcal{M}(d)}{d} \left(d + \log \|B\|\right)\right] \text{bit operations.}$$

As a consequence, the total cost of Algorithm 4 is (using the fact that the product over the loop iterations of the $\frac{\|\mathbf{b}_\kappa^{(old)}\|}{\|\mathbf{b}_\kappa^{(new)}\|}$'s is exactly $\prod_i \frac{\|\mathbf{b}_i^b\|}{\|\mathbf{b}_i^e\|}$):

$$\begin{aligned} &O\left[\sum_{\text{iterations}} \left(d + \log \frac{\|\mathbf{b}_\kappa^{(old)}\|}{\|\mathbf{b}_\kappa^{(new)}\|}\right) \frac{n\mathcal{M}(d)}{d} \left(d + \log \|B\|\right)\right] \\ =&O\left[\left(d + \log \prod \frac{d_i^b}{d_i^e} + \frac{1}{d} \log \prod \frac{\|\mathbf{b}_i^b\|}{\|\mathbf{b}_i^e\|}\right) n\mathcal{M}(d)(d + \log \|B\|)\right]. \end{aligned}$$

Since $\prod \|\mathbf{b}_i^b\| \leq \|B\|^d$ and $\prod d_i^b \leq \|B\|^{d^2}$, that bound immediately gives a $O(nd^2\mathcal{M}(d) \log \|B\|(d + \log \|B\|))$ complexity upper bound. $\square$

5. CONCLUSION

The decision to use Householder's transformations instead of Cholesky's factorization within LLL leads to modifications in the proof of correctness: the perturbations induced on the approximate R-factor have a different structure than in the L^2 algorithm of [9]. These modifications may probably be used for other forms or applications of the floating-point reduction of lattices. For example the new approach may be carried over to the case of linearly dependent input vectors, and to the case of stronger reductions (such as the fp Hermite-Korkine-Zolotarev reduction algorithm of [11]). An important direction that deserves to be investigated would be to try to further decrease the precision of the approximate computations. We showed that a precision essentially equal to the problem dimension is sufficient. Can we do better? It seems unnatural that a higher precision is required in H-LLL than in its (incomplete) underlying size-reduction algorithm. Finally, a more precise understanding of the numerical behavior is required for various aspects, such as the efficient implementation of H-LLL, which we are currently investigating.

ACKNOWLEDGMENTS. We thank the anonymous referees for their helpful comments. Ivan Morel and Damien Stehlé were partly funded by the LaRedA ANR project. Gilles Villard was partly funded by the Gecko ANR project.

6. REFERENCES

[1] X.-W. Chang, D. Stehlé, and G. Villard. Perturbation Analysis of the R-Factor of the QR Factorisation in the Context of LLL-Reduction. Work in progress, available at `http://perso.ens-lyon.fr/damien.stehle/QRPERTURB.html`, 2009.

[2] H. Cohen. *A Course in Computational Algebraic Number Theory, 2nd edition.* Springer, 1995.

[3] C. Hermite. Extraits de lettres de M. Hermite à M. Jacobi sur différents objets de la théorie des nombres, deuxième lettre. *J. reine angew Math*, 40:279–290, 1850.

[4] N. Higham. *Accuracy and Stability of Numerical Algorithms.* SIAM, 2002.

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

[6] H. Koy and C. P. Schnorr. Segment LLL-reduction of lattice bases with floating-point orthogonalization. In *Proc. of CALC'01*, volume 2146 of *LNCS*, pages 81–96. Springer, 2001.

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

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

[9] P. Nguyen and D. Stehlé. Floating-point LLL revisited. In *Proc. of Eurocrypt 2005*, volume 3494 of *LNCS*, pages 215–233. Springer, 2005. Extended version to appear in SIAM J. Comput., 2009.

[10] A. M. Odlyzko. The rise and fall of knapsack cryptosystems. In *Proc. of Cryptology and Computational Number Theory*, volume 42 of *Proc. of Symposia in Applied Mathematics*, pages 75–88. AMS, 1989.

[11] X. Pujol and D. Stehlé. Rigorous and efficient short lattice vectors enumeration. In *Proc. of Asiacrypt'08*, volume 5350 of *LNCS*, pages 390–405. Springer, 2008.

[12] C. P. Schnorr. Progress on LLL and lattice reduction. In *Proc. of the LLL+25 conference.* To appear in 2009.

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

[14] C. P. Schnorr. Fast LLL-type lattice reduction. *Inf. and Comp*, 204:1–25, 2006.

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

[16] D. Stehlé. Floating-point LLL: theoretical and practical aspects. In *Proc. of the LLL+25 conference.* To appear in 2009.

[17] A. Storjohann. Faster Algorithms for Integer Lattice Basis Reduction. Technical Report TR249, ETH-Zurich, Dpt. Comp. Sc., 1996.

[18] G. Villard. Certification of the QR factor R, and of lattice basis reducedness. In *Proc. ISSAC '07*, pages 361–368. ACM Press, 2007.