Segment and Strong Segment LLL-Reduction of Lattice Bases.

Henrik Koy[1] and Claus Peter Schnorr[2]

[1] Deutsche Bank AG, Frankfurt am Main, henrik.koy@db.com
[2] Fachbereiche Mathematik und Informatik, Universität Frankfurt, PSF 111932, D-60054 Frankfurt am Main, Germany. schnorr@cs.uni-frankfurt.de

April 22, 2002

Abstract. We present an efficient variant of LLL-reduction of lattice bases in the sense of LENSTRA, LENSTRA, LOVÁSZ [LLL82]. We organize LLL-reduction in segments of size k. Local LLL-reduction of segments is done using local coordinates of dimension $2k$.

Strong segment LLL-reduction yields bases of the same quality as LLL-reduction but the reduction is n-times faster for lattices of dimension n. We extend segment LLL-reduction to *iterated subsegments*. The resulting reduction algorithm runs in $O(n^3 \log n)$ arithmetic steps for integer lattices of dimension n with basis vectors of length $2^{O(n)}$, compared to $O(n^5)$ steps for LLL-reduction.

Keywords. LLL-reduction, shortest lattice vector, segments, iterated subsegments, local coordinates, local LLL-reduction.
Abbreviated Title. Segment and Strong Segment LLL.

1 Introduction.

The famous LLL-algorithm of LENSTRA, LENSTRA, LOVÁSZ [LLL82] for lattice basis reduction is a basic technique for solving important problems in algorithmic number theory, integer optimization, diophantine approximation and cryptography. Of the many possible applications we refer to a few recent ones [BN00, Bo00, Co98,NS00]. The LLL-algorithm requires $O(n^5)$ arithmetic steps on integers of bit length $O(n^2)$ when given n integer basis vectors of length $2^{O(n)}$ and dimension n. In this paper we improve the $O(n^5)$ bound to $O(n^3 \log n)$ using a novel LLL-type reduction.

We partition a basis $b_1, \dots, b_n$ of dimension $n = k\,m$ into m segments $b_{k(l-1)+1}$,..., b_{kl} of k consecutive basis vectors. LLL-exchanges are done locally in two consecutive segments using cordinates of dimension $2k$. Local LLL-exchanges cost merely $O(k^2)$ arithmetic steps, local size reduction included — compared to $O(n^2)$ steps for a global LLL-exchange.

First we introduce segment LLL-reduced bases, a variant of LLL-reduced bases that is designed to minimize the global overhead that complements local LLL-reductions. Segment LLL-reduction saves a factor n in the reduction time compared to LLL-reduction of lattices of dimension n. We present the basic concept of segment LLL-reduction in Section 3 and the strong version of it in Section 4. Strong segment LLL-reduction yields a basis where the first vector is as short as for LLL-bases. In Section 5 we present an even faster reduction algorithm using *iterated subsegments*. It has a proven time bound of $O(n^3 \log n)$ arithmetic steps for integer lattices of dimension n, given basis vectors of length $2^{O(n)}$. Section 6 contains the strong variant of segment LLL-reduction via iterated subsegments. Here, the first basis vector is as short as for LLL-bases.

The companion paper [KS01b] gives a practical implementation of segment LLL-reduction using floating point orthogonalization. Our present code reduces lattice bases of dimension 1000 consisting of integers of bit length 400 in 10 hours on a 800 MHz PC. Even for dimension $n < 100$ the new code is in practice much faster than previous codes. The use of iterated subsegments should further speed up the reduction in high dimensions.

Previously, Schönhage [Sc84] has used local coordinates to speed-up LLL-reduction. His concept of semi-reduction approximates the length of the shortest lattice vector up to a factor 2^n whereas we get close to a factor $(4/3)^{n/2}$. We use the [Sc84] analysis of the size of integers occuring in the reduction.

2 LLL-Reduction of Lattice Bases.

Notation. An ordered set of linearly independent vectors $b_1, ..., b_n \in \mathbf{Z}^d$ is a *basis* of the integer *lattice* $L = \sum_{i=1}^n b_i \mathbf{Z} \subset \mathbf{Z}^d$, consisting of all linear integer combinations of $b_1, ..., b_n$. We form the *basis matrix* $B = [b_1, ..., b_n] \in \mathbf{Z}^{d \times n}$ with column vectors $b_1, ..., b_n$, we write $L = L(B)$. Let $\widehat{b}_i$ denote the component of b_i that is orthogonal to $b_1, ..., b_{i-1}$ with respect to the *Euclidean inner product* $\langle x, y \rangle = x^\top y$. The *orthogonal vectors* $\widehat{b}_1, ..., \widehat{b}_n \in \mathbf{R}^d$ and the *Gram-Schmidt coefficients* $\mu_{j,i}$, $1 \le i, j \le n$, associated with the basis $b_1, ..., b_n$ satisfy for $j = 1, ..., n$

$$b_j = \textstyle\sum_{i=1}^j \mu_{j,i} \widehat{b}_i, \qquad \mu_{j,j} = 1, \qquad \mu_{j,i} = 0 \ \text{ for } \ i > j.$$

$$\mu_{j,i} = \langle b_j, \widehat{b}_i \rangle / \langle \widehat{b}_i, \widehat{b}_i \rangle, \qquad \langle \widehat{b}_j, \widehat{b}_i \rangle = 0 \text{ for } j \neq i.$$

Let $\|b\| = \langle b, b \rangle^{\frac{1}{2}}$ denote the *Euclidean length* of a vector $b \in \mathbf{R}^d$. A vector $b = \sum_{j=1}^n \mu_j \widehat{b}_j$ satifies $\|b\|^2 = \sum_{j=1}^n \mu_j^2 \|\widehat{b}_j\|^2$. Let λ_1 denote the length of the shortest non-zero lattice vector of a given lattice. The *determinant* of lattice $L = L(B)$ is

$$\det L = \det(B^\top B)^{\frac{1}{2}} = \textstyle\prod_{i=1}^n \|\widehat{b}_i\|.$$

Let $\pi_i : \mathbf{R}^n \to \mathrm{span}(b_1, ..., b_{i-1})^\perp$ denote the orthogonal projection, $\pi_i(b_k) = \sum_{j=i}^n \mu_{k,j} \widehat{b}_j$.

Definition 1. *A basis* $b_1, \ldots, b_n \in \mathbf{Z}^d$ *is an* LLL-basis (or LLL-reduced) *for given* $\delta \in]\frac{1}{4}, 1]$ *if*

1. $|\mu_{j,i}| \le \frac{1}{2}$ *for* $1 \le i < j \le n$,
2. $\delta \, \|\widehat{b}_i\|^2 \le \mu_{i+1,i}^2 \, \|\widehat{b}_i\|^2 + \|\widehat{b}_{i+1}\|^2$ *for* $i = 1, \ldots, n-1$.

LLL-bases have been introduced by A.K. LENSTRA, H.W. LENSTRA, JR. and L. LOVÁSZ [LLL82] who focused on $\delta = 3/4$. A basis with property 1. is called *size-reduced.* Extending [LLL82] to arbitrary $\delta \in]\frac{1}{4}, 1]$ and $\alpha := 1/(\delta - \frac{1}{4})$ yields Theorem 1. For the rest of the paper LLL-reduction refers to given δ, α.

Theorem 1. *An LLL-basis $b_1, \ldots, b_n$ of lattice L satisfies*

1. $\|b_1\|^2 \leq \alpha^{n-1} \lambda_1^2$ *and* $\|b_1\|^2 \leq \alpha^{i-1} \|\widehat{b}_i\|^2$ *for* $i = 1, \ldots, n$,
2. $\|b_1\|^2 \leq \alpha^{\frac{n-1}{2}} (\det L)^{\frac{2}{n}}$ *and* $\|\widehat{b}_n\|^2 \geq \alpha^{-\frac{n-1}{2}} (\det L)^{\frac{2}{n}}$.

Consider the QR-factorization $B = QR$ of the basis matrix $B = [b_1, \ldots, b_n] \in \mathbf{Z}^{d \times n}$, where $Q = [\,\widehat{b}_1/\|\widehat{b}_1\|, ..., \widehat{b}_n/\|\widehat{b}_n\|\,] \in \mathbf{R}^{d \times n}$ is an orthogonal matrix in the sense that $Q^\top Q = I_n$, and $R = [r_{i,j}]_{1 \leq i,j \leq n} = [\mathbf{r}_1, ..., \mathbf{r}_n] \in \mathbf{R}^{n \times n}$ is an upper-triangular matrix, $r_{i,j} = 0$ for $i > j$. We call R the *orthogonalization* of B. The transform $x \mapsto Qx$ preserves the inner product $\langle x, y \rangle = \langle Qx, Qy \rangle$, and thus R and B are isometrical basis matrices. We have that $\mu_{j,i} = r_{i,j}/r_{i,i}$, $|r_{i,i}| = \|\hat{b}_i\|$ and $\langle \mathbf{r}_i, \mathbf{r}_j \rangle = \langle b_i, b_j \rangle$. We present the core of the LLL-algorithm using the coefficients $r_{i,j}$. Clause 2 of Definition 1 means that $\delta\, r_{i,i}^2 \leq r_{i,i+1}^2 + r_{i+1,i+1}^2$ for $i = 1, ..., n-1$.

LLL-Algorithm (LLL)

INPUT $b_1, \ldots, b_n \in \mathbf{Z}^d,\ \delta$

OUTPUT $b_1, \ldots, b_n$ LLL-reduced basis

1. $l := 1$ (*l is the stage*)
2. WHILE $l \leq n$ DO

    compute $(r_{1,l}, ..., r_{l,l}, 0, ..., 0) = \|\widehat{b}_l\| \, (\mu_{l,1}, ..., \mu_{l,l}, 0, ..., 0)$

    size-reduce b_l against $b_{l-1}, ..., b_1$

    IF $l \neq 1$ AND $\delta\, r_{l-1,l-1}^2 > r_{l-1,l}^2 + r_{l,l}^2$

    THEN swap b_{l-1}, b_l, $l := l - 1$ ELSE $l := l + 1$.

More details. Size-reduction of b_l against $b_{l-1}, ..., b_1$ performs $b_l := b_l - \lceil \mu_{l,i} \rfloor b_i$ for $i = l-1, \ldots, 1$ where $\lceil r \rfloor = \lceil r - \frac{1}{2} \rceil$ denotes the nearest integer to $r \in \mathbf{R}$.

Each swap of b_{l-1}, b_l and each reduction step $b_l := b_l - \lceil \mu_{l,i} \rfloor b_i$ requires to update R. That update transforms two rows and two columns of R resulting in an upper-triangular R of the transformed basis, and can be done in $O(n^2)$ arithmetic steps. Therefore an LLL-exchange (swap of b_{l-1}, b_l) requires $O(nd)$ arithmetic steps, size-reduction of b_l and update of R included.

Local reduction. The LLL-algorithm reduces the 2×2-diagonal submatrices $\begin{bmatrix} r_{l-1,l-1} & r_{l-1,l} \\ 0 & r_{l,l} \end{bmatrix}$ of R, see Fig. 1. This amounts to a *local* LLL-reduction, where LLL-exchanges (swaps of b_{l-1}, b_l) are done in local coordinates of dimension 2. In Section 3 we study local LLL-reductions of diagonal $2k \times 2k$ submatrices of R using coordinates of dimension $2k$.

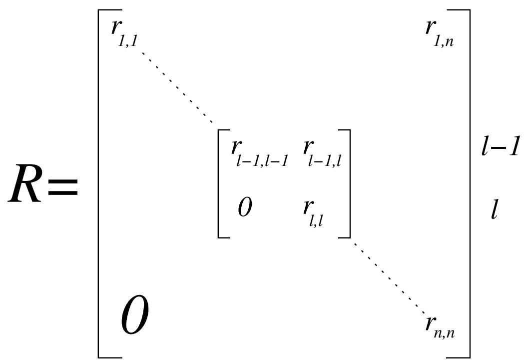

Fig. 1. The 2×2-diagonal submatrices of R

Integer arithmetic. As the coefficients $r_{i,j}$ of R are algebraic, integer arithmetic must instead use the rational numbers $\mu_{j,i}$, $\|\widehat{b}_i\|^2$. It is obvious how to replace in our algorithms the algebraic $r_{i,j}$ by rational numbers, and this does not affect the asymptotic bounds on the number of arithmetic steps.

Size of a basis. We measure the size of the basis $b_1, ..., b_n$ by

$$\overline{M} =_{\text{def}} \max_{i=1,...,n}(\|b_i\|^2, D_i),$$

where $D_i = \|\widehat{b}_1\|^2 \cdot ... \cdot \|\widehat{b}_i\|^2$ is the *Gramian determinant* of the sublattice with basis $b_1, ..., b_i$. Basis vectors of length $\|b_i\| = 2^{O(n)}$ satisfy $\overline{M} = 2^{O(n^2)}$, but $\overline{M} = 2^{O(n)}$ holds in many applications.

LLL-time bound. Given a basis $b_1, ..., b_n \in \mathbf{Z}^d$ of length $\max_i \|b_i\| = 2^{O(n)}$ and $d = O(n)$ the LLL-algorithm performs $O(n^5)$ arithmetic steps using $O(n^2)$-bit integers. These steps operate on the rational integers $\mu_{j,i}$, $\|\widehat{b}_i\|^2$ and the integer coordinates of the b_i. More precisely, numerators and denominators of these rationals have at most $O(n + \log_2 \overline{M})$ bits, see [LLL82,Sc84].

3 Segment LLL-Reduction.

Segments and local coordinates. Let the basis $b_1, \ldots, b_n \in \mathbf{Z}^d$ have dimension $n = k\,m$ and the QR-factorization $[b_1, \ldots, b_n] = B = QR$. We partition B into m *segments* $B_l = [b_{kl-k+1}, \ldots, b_{kl}]$ for $l = 1, ..., m$. Local LLL-reduction of two consecutive segments B_l, B_{l+1} is done in local coordinates of the submatrix

$$R_l := [r_{kl+i,kl+j}]_{-k<i,j\le k} \in \mathbf{R}^{2k\times 2k}$$

of R. R_l yields the local orthogonalization and the local coefficients $\mu_{kl+i,kl+j}$, $-k < i, j \le k$ of $[B_l, B_{l+1}]$. Global transforms complement local LLL-reductions. The novel concept of *segment LLL-reduction* (SLLL-reduction for short) minimizes the global overhead.

We let $D(l) = \|\widehat{b}_{k(l-1)+1}\|^2 \cdot ... \cdot \|\widehat{b}_{kl}\|^2$ denote the *local Gramian determinant* of segment B_l. We have that $D_{kl} = D(1) \cdot ... \cdot D(l)$.

Definition 2. *We call a basis* $b_1, \ldots, b_n \in \mathbf{Z}^d, n = km$*,* SLLL-basis (or SLLL-reduced) *for given* k, δ, α *if it is size-reduced and satisfies*

1. $\delta \|\widehat{b}_i\|^2 \leq \mu_{i+1,i}^2 \|\widehat{b}_i\|^2 + \|\widehat{b}_{i+1}\|^2$ *for* $i = 1, ..., n-1$ *except that* $i = 0 \mod k$*,*

2. $D(l) \leq (\alpha/\delta)^{k^2} D(l+1)$ *for* $l = 1, \ldots, m-1$*.*

The segments B_l of an SLLL-basis are LLL-reduced in the sense that the submatrix $[r_{kl+i,kl+j}]_{-k<i,j\leq 0} \in \mathbf{R}^{k\times k}$ of R is LLL-reduced. Clause 1 does not bridge distinct segments due since the i with $i = 0 \mod k$ are excepted. Property 2 is weaker than the property $D(l) \leq \alpha^{k^2} D(l+1)$ of LLL-bases which follows from Definition 1. This weakening is used to reduce the number of local LLL-reductions of R_l.

Property 2 of SLLL-bases is preserved under duality, if it holds for a basis $b_1, ..., b_n$ it also holds for the dual basis $b_1^*, ..., b_n^*$. The *dual* of lattice L is the lattice

$$L^* =_{\text{def}} \{x \in \text{span}(L) \mid \langle x, y\rangle \in \mathbf{Z} \text{ for all } y \in L\}.$$

We have that $\det L^* = (\det L)^{-1}$. The *dual* B^* of a basis matrix $B \in \mathbf{Z}^{n\times n}$, satisfying $L(B^*) = L(B)^*$, is constructed by inverting the order of the columns of the matrix $(B^{-1})^\top$, the transpose of the inverse of B. The *dual basis* $[b_1^*, ..., b_n^*] = [b_1, ..., b_n]^*$ satisfies $\langle b_i^*, b_j\rangle = \delta_{i,j}$ and $\|\widehat{b}_i\| = \|\widehat{b}_{n-i+1}^*\|^{-1}$ for $i = 1, ..., n$.

Theorem 2. *Every SLLL-basis* $b_1, ..., b_n$ *satisfies*

1. $\|b_1\|^2 \leq (\alpha/\delta)^{\frac{n-1}{2}} (\det L)^{\frac{2}{n}}$*,* *2.* $\|\widehat{b}_n\|^2 \geq (\delta/\alpha)^{\frac{n-1}{2}} (\det L)^{\frac{2}{n}}$*.*

Proof. 1. By definition of SLLL-reducedness we have that

$$D(1) \leq (\alpha/\delta)^{k^2(i-1)} D(i) \qquad \text{for } i = 1, ..., n.$$

As $D(1) \cdot ... \cdot D(m) = (\det L)^2$ and $1 + 2 + ... + m - 1 = m\frac{m-1}{2}$ this yields

$$D(1) \leq (\alpha/\delta)^{k^2 \frac{m-1}{2}} (\det L)^{\frac{2}{m}}.$$

Moreover $\|b_1\|^2 \leq \alpha^{\frac{k-1}{2}} D(1)^{\frac{1}{k}}$ holds by Theorem 1 as the basis $b_1, ..., b_k$ is LLL-reduced. The two latter inequalities imply the claim

$$\|b_1\|^2 \leq \alpha^{\frac{k-1}{2}} (\alpha/\delta)^{k\frac{m-1}{2}} (\det L)^{\frac{2}{mk}} \leq (\alpha/\delta)^{\frac{n-1}{2}} \det L^{\frac{2}{n}}.$$

2. Clause 2 of Definition 2 and Theorem 1 also hold for the dual basis $b_1^*, ..., b_n^*$ of the dual lattice. We have that $\|b_1^*\| = \|\widehat{b}_n\|^{-1}$ and $\det(L^*) = (\det L)^{-1}$. Applying the proof of Inequality 1 to the dual basis $b_1^*, ..., b_n^*$ yields Inequality 2. □

Algorithm for SLLL-reduction. The algorithm **SLLL** transforms a given basis into an SLLL-basis. It iterates local LLL-reduction of two segments $[B_l, B_{l+1}] = [b_{kl-k+1}, ..., b_{kl+k}]$ via **loc-LLL**(l). **SLLL** emulates the LLL-algorithm replacing the vector r_{l-1} by the segment B_l.

Segment LLL (SLLL)

INPUT $\quad b_1, \ldots, b_n \in \mathbf{Z}^d, k, m, n = km,\ \delta$

OUTPUT $\quad b_1, \ldots, b_n \quad$ SLLL-basis

1. $l := 1$, compute the orthogonalization $R \in \mathbf{R}^{n \times n}$
2. WHILE $l \leq m - 1$ DO

 loc-LLL(l) (LLL-reduces R_l)

 IF $l > 1$ AND $D(l-1) > (\alpha/\delta)^{k^2} D(l)$

 THEN $l := l - 1$ ELSE $l := l + 1$.

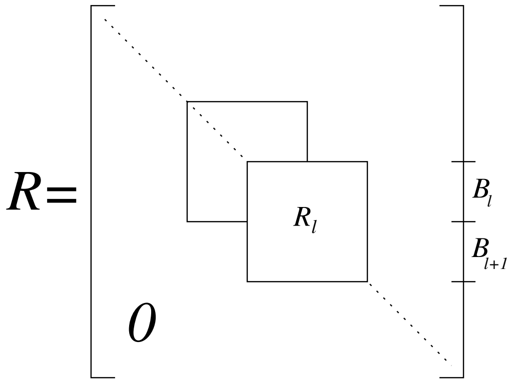

Fig. 2. Areas of subsequent local LLL-reductions.

loc-LLL(l)

Given are the orthogonalization $R \in \mathbf{R}^{n \times n}$ and the submatrix $R_l \in R^{2k \times 2k}$.

1. *local LLL-reduction.* First size-reduce then LLL-reduce R_l. After each reduction step update R_l into upper-triangular form, record the basis transform of R_l in the matrix $H_l \in \mathbf{Z}^{2k \times 2k}$. (The basis transform operates on R_l from the right, the update into upper-triangular form operates from the left.)
2. $[B_l, B_{l+1}] := [B_l, B_{l+1}]\, H_l$, reset H_l, update R into upper-triangular form.
3. Size-reduce $[B_l, B_{l+1}]$ globally and update R accordingly.

LLL-exchanges in the local LLL-reduction of Step 1 are done in local coordinates of dimension $2k$. A local LLL-exchange merely requires $O(k^2)$ arithmetic steps, update of R_l and local size-reduction included. Compare this to the $O(nd)$ arithmetic steps for an LLL-exchange in global coordinates. Steps 2, 3 of **loc-LLL**(l) perform $O(ndk)$ arithmetic steps.

SLLL generates bases that are slightly better than expressed by the inequalities of Theorem 2. Upon termination of **loc-LLL**(l) we have that $D(l) \leq \alpha^{k^2} D(l+1)$ while Theorem 2 merely assumes $D(l) \leq (\alpha/\delta)^{k^2} D(l+1)$.

The number of **loc-LLL** *executions.* The Lovász volume argument shows that the number of executions of **loc-LLL** decreases cubically in k. We let dec denote the number of times that l decreases due to $D(l-1) > (\alpha/\delta)^{k^2} D(l)$.

Theorem 3. *The number of* **loc-LLL** *executions in* **SLLL** *is* $m - 1 + 2 \cdot \mathrm{dec}$, *where* $\mathrm{dec} \leq 2n\, k^{-3} \log_{1/\delta} \overline{M}$.

Proof. Each **loc-LLL** execution resulting in $l := l-1$ is compensated by another one resulting in $l := l+1$. There are $m-1$ additional executions to proceed from $l = 1$ to $l = m-1$. Hence, it remains to bound dec.

We show that a **loc-LLL**$(l-1)$ execution due to $D(l-1) > (\alpha/\delta)^{k^2} D(l)$ decreases $D(l-1)$ by the factor $\delta^{k^2/2}$. **loc-LLL**$(l-1)$ performs a local LLL-reduction of R_l, it changes $D(l-1), D(l)$ into $D'(l-1), D'(l)$ but preserves $D(l^*)$ for $l^* \neq l, l+1$. It also preserves the product $D(l-1)D(l)$. As the local LLL-reduction yields $D'(l-1) \leq \alpha^{k^2} D'(l)$ we have that

$$\begin{aligned} D'(l-1) &\leq \alpha^{k^2} D'(l) = \alpha^{k^2} D'(l-1)D'(l)/D'(l-1) \\ &= \alpha^{k^2} D(l-1)D(l)/D'(l-1) < \delta^{k^2} D(l-1)^2/D'(l-1), \end{aligned}$$

and thus $D'(l-1) \leq \delta^{k^2/2} D(l-1)$. Hence **loc-LLL**$(l)$ decreases $\mathbf{D} =_{def} \prod_{j=1}^{m-1} D_{jk}$ by the factor $\delta^{k^2/2}$. As $\mathbf{D}$ is a positive integer, $\mathbf{D} \leq \overline{M}^{m-1}$, this implies

$$\mathrm{dec} \leq \log_{1/\delta^{k^2/2}} \overline{M}^{m-1} \leq 2\tfrac{m-1}{k^2} \log_{1/\delta} \overline{M}. \qquad \square$$

The number of LLL-exchanges. **LLL** corresponds to **SLLL** with $k = 1$. However, an LLL-exchange decreases $\mathbf{D}$ by a factor δ and not just by $\sqrt{\delta}$ as in the above proof. Hence, the number of swaps in the **LLL** is at most $(n-1)\log_{1/\delta} \overline{M}$, the factor 2 in Theorem 3 disappears.

Theorem 4. *Let* $k = \Theta(m) = \Theta(\sqrt{n})$. *Then* **SLLL** *performs* $O(nd \log_{1/\delta} \overline{M})$ *arithmetic steps using integers of bit length* $O(n + \log_2 \overline{M}) = O(n^2)$.

SLLL improves the LLL-time bound from $O(n^2 d \log_{1/\delta} \overline{M})$ to $O(nd \log_{1/\delta} \overline{M})$ arithmetic steps, saving a factor n.

Proof. Time bound. We separately count the *local* (resp. *global*) arithmetic steps in Step 1 (resp. Steps 2, 3) of **loc-LLL**(l). There are at most $n \log_{1/\delta} \overline{M}$ LLL-exchanges, done in local coordinates of dimension $2k$, each requiring $O(k^2)$ steps for local size-reduction, for updating R_l into upper-triangular form and for updating H_l. In total there are $O(nk^2 \log_{1/\delta} \overline{M})$ local arithmetic steps.

Each execution of **loc-LLL** requires $O(ndk)$ global arithmetic steps for updating R into upper-triangular form, global size-reduction, and segment transform in Steps 2, 3 of **loc-LLL**(l). The initial computation of R requires $O(n^2 d)$ arithmetic steps. By Theorem 3 there are $O(n^2 d + n^2 d + m^2 d \log_{1/\delta} \overline{M})$ global arithmetic steps.

The choice $k, m = \Theta(\sqrt{n})$ equalizes for $d = O(n)$ these bounds for the local and global arithmetic steps. In total there are at most $O(nd \log_{1/\delta} \overline{M})$ local and global arithmetic steps. [1]

Size of the integers. Recall that the determinants D_i do not increase during LLL-reduction. In particular, we always have that $1 \leq D_i \leq \overline{M}$, and $\|\widehat{b}_i\|^2 = D_i/D_{i+1}$ is a rational integer with numerator and denominator bounded by $\overline{M}$, $\overline{M}^{-1} \leq \|\widehat{b}_i\|^2 \leq \overline{M}$.

We next show that $\max_i \|b_i\|^2$ can temporarily increase not more than by a factor $2^{O(n)}\overline{M}$. We separately study how size-reduction and local LLL-reduction affect $\max_i \|b_i\|^2$.

The length $\|b_i\|$ can temporarily increase during size-reduction of b_i according to $b_i := b_i - \lceil \mu_{i,j} \rfloor b_j$ for $j = i-1, ..., 1$. A step $b_i := b_i - \lceil \mu_{i,j} \rfloor b_j$ induces $\mu_{i,h} := \mu_{i,h} - \lceil \mu_{i,j} \rfloor \mu_{j,h}$ for $h = 1, ..., j$. As $b_1, ..., b_{i-1}$ are size-reduced we have that $|\mu_{j,h}| \leq \frac{1}{2}$ for $1 \leq h < j$. Hence, a size-reduction step increases $M_i := \max_{h<i} |\mu_{i,h}|$ by at most a factor $\frac{3}{2}$, and $\|b_i\|$ can temporarily increase during size-reduction of b_i not more than by a factor $(\frac{3}{2})^{i-1}$.

Consider the coefficients of the matrix $H \in \mathbf{Z}^{2k \times 2k}$ of the local LLL-reduction that transforms segments B_l, B_{l+1} according to $[B_l, B_{l+1}] := [B_l, B_{l+1}]H$. We let $b'_j, \widehat{b}'_j, \mu'_{j,i}$ denote the values of the transformed segments $[b'_{kl-k+1}, ..., b'_{kl+k}] = [B_l, B_{l+1}]H$. We let $\|H\|_1$ denote the maximal $\| \ \|_1$-norm of the columns of H.

Lemma 1. [Sc84, Inequality (3.3)] *We have that*

1. $H = ([\mu_{j,i}]^\top)^{-1} \ [\langle \widehat{b}_i, \widehat{b}'_j \rangle \|\widehat{b}_i\|^{-2}]_{kl-k<i,j\leq kl+k} \ [\mu'_{j,i}]^\top$,
2. $\|H\|_1 \leq (2k)^2 (\frac{3}{2})^{2k-1} \overline{M} \leq 2^{O(k)} \overline{M}$.

Proof. Equality 1. follows from the equations

$$\begin{aligned}[b'_{kl-k+1}, ..., b'_{kl+k}] &= [\widehat{b}'_{kl-k+1}, ..., \widehat{b}'_{kl+k}][\mu'_{j,i}]^\top \\ &= [\widehat{b}_{kl-k+1}, ..., \widehat{b}_{kl+k}][\mu_{j,i}]^\top H.\end{aligned}$$

The segments B_l, B_{l+1} are already size-reduced when starting the local LLL-reduction of R_l, i.e., $|\mu_{j,i}| \leq \frac{1}{2}$ for $kl - k < i < j \leq kl + k$. Then the coefficients $\nu_{j,i}$ of the inverse matrix $[\nu_{j,i}] = [\mu_{j,i}]^{-1}$ satisfy $|\nu_{j,i}| \leq (\frac{3}{2})^{|j-i|}$. Inequality 2. follows from 1. as $|\langle \widehat{b}_i, \widehat{b}'_j \rangle \|\widehat{b}_i\|^{-2}| \leq \|\widehat{b}'_j\| / \|\widehat{b}_i\| \leq \overline{M}$ and $|\mu'_{j,i}| \leq \frac{1}{2}$ for $i < j$. □

Conclusion. All integers arising in **SLLL**-execution are bounded in absolute value by $2^{O(n)}\overline{M}^{3/2}$ having bit length $O(n + \log_2 \overline{M})$. In particular the vectors b_i of $[B_l, B_{l+1}]H$ in **loc-LLL**(l) satisfy by Lemma 1 that $\|b_i\| \leq \overline{M}^{1/2} \|H\|_1 = 2^{O(k)}\overline{M}^{3/2}$. The final size-reduction of b_i in **loc-LLL**(l) can temporarily increase $\|b_i\|$ by a factor $(\frac{3}{2})^{i-1}$. The size-reduced b_i satisfies $\|b_i\|^2 \leq \sum_{j=1}^{i} \mu_{i,j}^2 \|\widehat{b}_i\|^2 \leq \frac{i+1}{4}\overline{M}$. That bound holds after each execution of **loc-LLL**(l). □

[1] In practice, the global steps get dominant for $k \gg m$, not yet for $k \approx m$. This is because the local steps operate on smaller integers. In the [KS01b] implementation, these steps are in fast floating point arithmetic. Even segment sizes as large as $k = 100$ yield good running times.

Dependence of time bounds on δ. The time bounds contain a factor $\log_{1/\delta} 2$,

$$\log_{1/\delta} 2 = \log_2(e)/\ln(1/\delta) \leq \log_2(e)\frac{\delta}{1-\delta},$$

since $\ln(1/\delta) \geq 1/\delta - 1$. We see that replacing δ by $\sqrt{\delta}$ essentially halves $1 - \delta$ and doubles the SLLL-time bound. In practice, the reduction time may increase slower than by the factor $\frac{\delta}{1-\delta}$ as δ approaches 1, see [KS01b, Fig.3] comparing reduction times for $\delta = 0.99$ and $\delta = 0.999$.

4 Strong Segment LLL-Reduction.

We strengthen SLLL-bases as to satisfy $\|b_1\|^2 \leq \lambda_1^2\,(\alpha/\delta)^{n-1}$. This comes close to the property $\|b_1\|^2 \leq \lambda_1^2\,\alpha^{n-1}$ of LLL-bases in Theorem 1. Recall that k, δ, α refer to SLLL-reduction.

Notation. We call segment $B_l = [b_{kl-k+1}, ..., b_{kl}]$ of lattice basis $b_1, \ldots, b_n$ *strong* if

$$\|b_1\|^2/\|\widehat{b}_i\|^2 \leq (\alpha/\delta)^{i-1} \quad \text{for } i = kl-k+1, \ldots, kl,$$

otherwise B_l is called *weak*. B_l is called *completely weak* if

$$\|b_1\|^2/\|\widehat{b}_i\|^2 > (\alpha/\delta)^{i-1} \quad \text{for } i = kl-k+1, ..., k\,l.$$

We call the basis $b_1, \ldots, b_n$ *strong* if all segments are strong. We call the index i *weak* if $\|b_1\|^2/\|\widehat{b}_i\|^2 > (\alpha/\delta)^{i-1}$.

Lemma 2. *A strong basis $b_1, ..., b_n$ satisfies $\|b_1\|^2 \leq \lambda_1^2\,(\alpha/\delta)^{n-1}$.*

Proof. Every basis $b_1, ..., b_n$ satisfies $\lambda_1 \geq \max_i \|\widehat{b}_i\|$. This implies that $\|b_1\|^2 \leq \lambda_1^2 \max_i \|b_1\|^2/\|\widehat{b}_i\|^2$. This proves the claim for a strong basis. □

Strong SLLL-bases versus LLL-bases. As LLL-bases satisfy $\|b_1\|^2 \leq \lambda_1^2\,\alpha^{n-1}$ we compare α/δ with α. LLL-bases are better than strong SLLL-bases for the same δ. However, strong SLLL-bases for $\delta' = \sqrt{\delta}$ are better than LLL-bases for δ, because $\alpha'/\delta' = \frac{1}{(\delta')^2-\delta'/4} < \frac{1}{\delta-1/4} = \alpha$.

Strong SLLL-bases with $\delta' = 0.95$ are better than LLL-bases with $\delta = 0.9$. Replacing δ by $\sqrt{\delta}$ increases the SLLL-time bound at most by a factor 2.

Strong SLLL (SSLLL)

INPUT SLLL-basis $b_1, \ldots, b_n \in \mathbf{Z}^d, n = km$

OUTPUT $b_1, \ldots, b_n$ strong SLLL-basis

1. IF all segments are strong THEN size-reduce $b_1, ..., b_n$ and terminate
2. **loc-LLL**(l) for the maximal l such that B_{l+1} is weak
3. WHILE $D(l) > (\alpha/\delta)^{k^2} D(l+1)$ for some l DO **loc-LLL**(l)
4. GO TO 1.

Time analysis. **SSLLL** iterates two types of **loc-LLL**(l) executions:

- executions in Step 2 due to a weak B_{l+1}.
- executions in Step 3 due to $D(l) > (\alpha/\delta)^{k^2} D(l+1)$.

Executions in Step 2 are done only when $D(l) \le (\alpha/\delta)^{k^2} D(l+1)$ holds for all l. We show in Lemma 3 below that these **loc-LLL**(l) executions either decrease

$$\text{maw} =_{\text{def}} \max\{l \mid B_l \text{ is weak}\},$$

or else there directly follows an execution of **loc-LLL** in Step 3. Clearly, maw cannot increase during **SSLLL** execution since $\max_{i=1,\dots,n}\{i \mid i \text{ weak}\}$ cannot increase during LLL-reduction. Hence maw decreases at most $m-1$ times.

Recall that an execution in Step 3 decreases $\mathbf{D} := \prod_{k=1}^{m-1} D_{lk}$ by a factor $\delta^{k^2/2}$ while an execution in Step 2 does not increase $\mathbf{D}$. Thus, the number of executions in Step 3 is at most dec $\le 2\,nk^{-3}\log_{1/\delta}\overline{M}$. As maw decreases at most $m-1$ times we see that

$$\#\text{executions in Step 2} \le m + \#\text{ executions in Step 3} \le m + 2\,nk^{-3}\log_{1/\delta}\overline{M}.$$

This shows that the total number of **loc-LLL** executions in **SSLLL** is at most $m + 4\,nk^{-3}\log_{1/\delta}\overline{M}$. This proves the following Theorem.

Theorem 5. *The* **SLLL***-time bound in Theorem 4 also holds for* **SSLLL***.*

It remains to prove

Lemma 3. *If* **loc-LLL**(l) *is executed in Step 2 of* **SSLLL** *then either the resulting* B_{l+1} *is strong or else the resulting* B_l *is completely weak, in which case we have that* $D(l-1) > (\alpha/\delta)^{k^2} D(l)$.

We see that under the assumption of Lemma 3 either maw decreases or else **loc-LLL**(l) is directly followed by a **loc-LLL** execution in Step 3. This is the main argument in the proof of Theorem 5.

Proof. Suppose that after executing **loc-LLL**(l) there is a weak $j = kl + i$ of B_{l+1}, $1 \le i \le k$. Then *all* $kl + i'$ are weak for $-k < i' \le i$. This holds because $[B_{l+1}, B_l]$ is locally LLL-reduced and thus $\|\widehat{b}_{kl+i}\|^2 \le \alpha\,\|\widehat{b}_{kl+i+1}\|^2$ holds for $-k < i < k$.

We conclude that the B_l resulting from **loc-LLL**(l) is completely weak if the resulting B_{l+1} is weak. We finally show that $D(l-1) > (\alpha/\delta)^{k^2} D(l)$ holds in this case.

As $b_1, \dots, b_{kl-k}$ is SLLL-reduced we have that

$$\|b_1\|^{2k} \le \alpha^{\binom{k}{2}} D(1) \qquad \text{(since } b_1, \dots, b_k \text{ is LLL reduced)}$$

$$D(1) \le (\alpha/\delta)^{k^2(l-2)}\, D(l-1), \qquad \text{and thus}$$

$$\|b_1\|^{2k} \le (\alpha/\delta)^{k^2(l-2)}\, \alpha^{\binom{k}{2}} D(l-1).$$

On the other hand, as B_l is completely weak, we have that

$$\|b_1\|^{2k} > (\alpha/\delta)^{k^2(l-1)+\binom{k}{2}}\, D(l).$$

The latter two inequalities imply that $D(l-1) > (\alpha/\delta)^{k^2}(1/\delta)^{\binom{k}{2}}\, D(l)$, proving the claim. $\square$

5 Reduction via Iterated Subsegments.

We extend the concept of segment LLL-reduction to an iterative structure of segments of levels $\sigma = 0, 1, ..., s$. Segments of level σ partition into segments of level $\sigma - 1$. We extend the concept of SLLL-bases to ISLLL-bases in that we relax the inequality $D(l) \le (\alpha/\delta)^{k^2} D(l+1)$ to $D(l) \le (\alpha/\delta^\sigma)^{k^2} D(l+1)$ for segments of level σ and size k. This relaxation will furher reduce the number of local reductions of the large segments of high level.

Let $n = k_1 \cdot ... \cdot k_s$ be a product of integers $k_1, \ldots, k_s \ge 2$, $s \le \log_2 n$. For given $\mathbf{k} = (k_1, ..., k_s)$ we denote $\mathbf{k}_\sigma := k_1 \cdot ... \cdot k_\sigma$ for $\sigma = 1, \ldots, s$, $\mathbf{k}_0 := 1$. We use segments of level $\sigma = 0, ..., s-1$, the *segments of level* σ are

$$B_l^{(\sigma)} = [b_{\mathbf{k}_\sigma(l-1)+1}, ..., b_{\mathbf{k}_\sigma l}] \qquad \text{for } l = 1, ..., n/\mathbf{k}_\sigma.$$

Segment $B_l^{(\sigma)}$ has size $\mathbf{k}_\sigma$ and partitions into k_σ segments of level $\sigma - 1$, $B_l^{(\sigma)} = [B_{k_\sigma(l-1)+1}^{(\sigma-1)}, ..., B_{k_\sigma l}^{(\sigma-1)}]$. Local reduction of $[B_l^{(\sigma)}, B_{l+1}^{(\sigma)}]$ is done in the coordinates of

$$R_l^{(\sigma)} =_{\mathrm{def}} [r_{\mathbf{k}_\sigma l+i, \mathbf{k}_\sigma l+j}]_{-\mathbf{k}_\sigma < i,j \le \mathbf{k}_\sigma} \in \mathbf{R}^{2\mathbf{k}_\sigma \times 2\mathbf{k}_\sigma}.$$

The submatrix $R_l^{(\sigma)}$ of R yields the local orthogonalization and local Gram-Schmidt coefficients of $[B_l^{(\sigma)}, B_{l+1}^{(\sigma)}]$.

Informal argument for the $O(n^3 \log n)$ time bound. We improve the SLLL-time bound by using the step bounds $O(k^2 n \log_{1/\delta} \overline{M})$ and $O(m^2 d \log_{1/\delta} \overline{M})$ for the local and global steps only for $k = k_1$ and $m = k_2, ..., k_s$. Global transforms extend local transforms from level $\sigma - 1$ to level σ. They corresponds to global steps in **SLLL**, and the **SLLL**-bound for the number of global SLLL-steps applies with $m = k_\sigma$. As the global steps are done for all levels we get an additional time factor $s \le \log_2 n$. If $\max_\sigma k_\sigma = O(1)$, $d = O(n)$ and $\overline{M} = 2^{O(n^2)}$ there are in total $O(n^3\, s)$ arithmetic steps.

We let $D^{(\sigma)}(l) = \|\widehat{b}_{\mathbf{k}_\sigma(l-1)+1}\|^2 \cdots \|\widehat{b}_{\mathbf{k}_\sigma l}\|^2$ denote the *local determinant* of the segment $B_l^{(\sigma)}$, $D^{(0)}(l) = \|\widehat{b}_l\|^2$.

Definition 3. *A basis $b_1, \ldots, b_n \in \mathbf{Z}^d, n = k_1 \cdots k_s = \mathbf{k}_s$ is an* ISLLL-basis *for given $\mathbf{k}, \delta, \alpha$ if it is size-reduced and satisfies*

$$D^{(\sigma)}(l) \le (\alpha/\delta^\sigma)^{(\mathbf{k}_\sigma)^2} D^{(\sigma)}(l+1) \tag{1}$$

for $\sigma = 0, ..., s-1$ and $l = 1, \ldots, n/\mathbf{k}_\sigma - 1$, except that $l = 0 \mod k_{\sigma+1}$.

Since $l = 0 \mod k_{\sigma+1}$ is excepted in (1) these conditions hold for constant σ locally in segments $B_l^{(\sigma+1)}$ of level $\sigma + 1$, they do not bridge distinct such segments. The conditions (1) can be written for $\sigma = 0$ as

$$\|\widehat{b}_l\|^2 \leq \alpha \, \|\widehat{b}_{l+1}\|^2 \qquad \text{for } l \neq 0 \mod k_1.$$

If δ is close to 1 so is δ^σ because $\sigma \leq \log_2 n$. For $n = k_1 \cdot k_2 = k \cdot m$, $s = 2$, Definition 3 repeats Definition 2 slightly weakening Clause 1.

ISLLL-reducedness is preserved under duality. If the basis $b_1, ..., b_n$ is ISLLL-reduced so is the dual basis $b_1^*, ..., b_n^*$.

We next extend Theorem 2 to iterated segments.

Theorem 6. *Every ISLLL-basis $b_1, \ldots, b_n$, $n = k_1 \cdots k_s = \mathbf{k}_s$ satisfies $\|b_1\|^2 \leq (\alpha/\delta^{s-1})^{\frac{n-1}{2}} (\det L)^{\frac{2}{n}}$ and $\|\widehat{b}_n\|^2 \geq (\delta^{s-1}/\alpha)^{\frac{n-1}{2}} (\det L)^{\frac{2}{n}}$.*

Proof. We prove by induction on σ that

$$\|b_1\|^{2\,\mathbf{k}_\sigma} \leq (\alpha/\delta^{\sigma-1})^{\mathbf{k}_\sigma\,(\mathbf{k}_\sigma-1)/2} \; D^{(\sigma)}(1).$$

For $\sigma = s$, this proves the first claim of the theorem because $D^{(s)}(1) = (\det L)^2$, $\mathbf{k}_s = n$. The second claim follows by duality.

The induction hypothesis is trivial for $\sigma = 0$ as we have that $(\mathbf{k}_0 - 1)/2 = 0$, $\mathbf{k}_0 = 1$ and $D^{(0)}(1) = \|b_1\|^2$.

Induction from σ to $\sigma + 1$. By ISLLL-reducedness we have that

$$D^{(\sigma)}(1) \leq (\alpha/\delta^\sigma)^{(\mathbf{k}_\sigma)^2(l-1)} \; D^{(\sigma)}(l) \quad \text{for } l = 1, \ldots, k_{\sigma+1}.$$

Using the equation $D^{(\sigma+1)}(1) = \prod_{l=1}^{k_{\sigma+1}} D^{(\sigma)}(l)$ and $\sum_{l=1}^{k_{\sigma+1}} (l-1) = \binom{k_{\sigma+1}}{2}$ this yields

$$D^{(\sigma)}(1)^{k_{\sigma+1}} \leq (\alpha/\delta^\sigma)^{(\mathbf{k}_\sigma)^2 \binom{k_{\sigma+1}}{2}} \; D^{(\sigma+1)}(1).$$

Using the induction claim for σ and $\mathbf{k}_{\sigma+1} = \mathbf{k}_\sigma k_{\sigma+1}$this yields

$$\|b_1\|^{2\,\mathbf{k}_{\sigma+1}} \leq (\alpha/\delta^{\sigma-1})^{\binom{\mathbf{k}_\sigma}{2} k_{\sigma+1}} \; (\alpha/\delta^\sigma)^{(\mathbf{k}_\sigma)^2 \binom{k_{\sigma+1}}{2}} \; D^{(\sigma+1)}(1)$$

Hence the claim for $\sigma + 1$ since $\binom{\mathbf{k}_\sigma}{2} k_{\sigma+1} + (\mathbf{k}_\sigma)^2 \, \binom{k_{\sigma+1}}{2} = \mathbf{k}_{\sigma+1} \frac{\mathbf{k}_{\sigma+1}-1}{2}$. $\square$

ISLLL

INPUT $\quad b_1, \ldots, b_n \in \mathbf{Z}^d$, $n = k_1 \cdot \ldots \cdot k_s = \mathbf{k}_s$

OUTPUT $\quad b_1, \ldots, b_n$ ISLLL-basis

1. $l := 1$, compute $R \in \mathbf{R}^{n \times n}$
2. WHILE $\quad l \leq k_{s-1} - 1 \quad$ DO

   loc-ISL$^{(s-1)}(l)$ (ISLLL-reduces $R_l^{(s-1)}$)

   IF $l > 1$ AND $D^{(s-1)}(l-1) > (\alpha/\delta^{s-1})^{(\mathbf{k}_{s-1})^2} D^{(s-1)}(l)$

   THEN $l := l - 1$ ELSE $l := l + 1$.

Algorithm for ISLLL-reduction. The algorithm **ISLLL** transforms a given basis of dimension $n = \mathbf{k}_s$ into an ISLLL-basis. It iteratively performs local ISLLL-reductions of local R-matrices $R_l^{(s-1)}$ by the procedure **loc-ISL**$^{(s-1)}(l)$. ISLLL-reduction of $R_l^{(s-1)}$ transforms the basis matrix $R_l^{(s-1)}$ from the right and updates it into upper-triangular form. Upon termination of **loc-ISL**$^{(s-1)}(l)$

$$D^{(\sigma)}(l) \le (\alpha/\delta^{\sigma})^{(\mathbf{k}_\sigma)^2} D^{(\sigma)}(l+1)$$

holds for all subdeterminants $D^{(\sigma)}(l)$, $D^{(\sigma)}(l+1)$ of $R_l^{(s-1)}$, i.e., for $\mathbf{k}_{s-1}(l-1) \le \mathbf{k}_\sigma(l-1)$ and $\mathbf{k}_\sigma(l+1) \le \mathbf{k}_{s-1}(l+1)$. The procedure **loc-ISL**$^{(s-1)}(l)$ iteratively ISLLL-reduces submatrices $R_{l'}^{(\sigma)}$ of level $\sigma < s-1$ via the procedure **loc-ISL**$^{(\sigma)}(l')$.

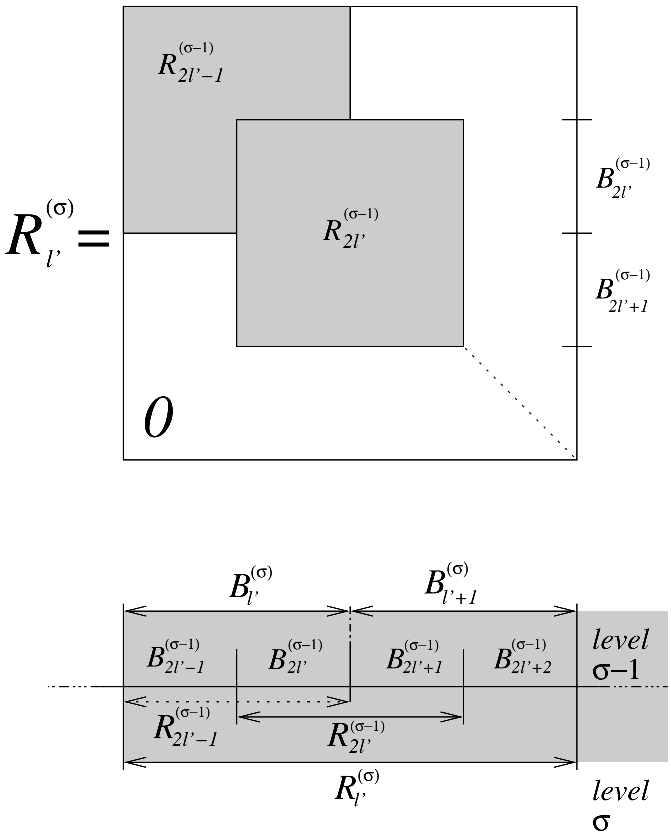

Fig. 3. Iterative segmentsw of levels $\sigma - 1$ and σ for $k_\sigma = 2$.

The interaction between levels $\sigma-1$ and σ. Consider two segments $[B_l^{(\sigma-1)}, B_{l+1}^{(\sigma-1)}] \subset [B_{l'}^{(\sigma)}, B_{l'+1}^{(\sigma)}]$ for $l' := \lceil l/k_\sigma \rfloor$. By the centered choice of l' the matrices $R_{l-1}^{(\sigma-1)}$,

$R_l^{(\sigma-1)}$, $R_{l+1}^{(\sigma-1)}$ are all covered by $R_{l'}^{(\sigma)}$, see Fig. 3. As the areas of $R_l^{(\sigma-1)}$, $R_{l\pm1}^{(\sigma-1)}$ overlap, a basis transform $H_l^{(\sigma-1)}$ of $R_l^{(\sigma-1)}$ must first be "transported" to level σ before $R_{l\pm1}^{(\sigma-1)}$ can be locally reduced. A subsequent reduction of $R_{l\pm1}^{(\sigma-1)}$ by **loc-ISL**$^{(\sigma-1)}(l\pm1)$ starts by copying $R_{l\pm1}^{(\sigma-1)}$ from the updated $R_{l'}^{(\sigma)}$.

Let $H_l^{(\sigma-1)} \in \mathbf{Z}^{2\mathbf{k}_{\sigma-1}\times 2\mathbf{k}_{\sigma-1}}$ denote the basis transform that has been done on $R_l^{(\sigma-1)}$ and has not yet been "transported" to level σ.

Transporting the basis transform $H_l^{(\sigma-1)}$ to level σ means to transform $R_{l'}^{(\sigma)}$ and $H_{l'}^{(\sigma)}$ as follows: multiply the submatrices of $2\mathbf{k}_{\sigma-1}$ columns of $R_{l'}^{(\sigma)}$ and $H_{l'}^{(\sigma)}$ corresponding to $[B_l^{(\sigma-1)}, B_{l+1}^{(\sigma-1)}]$ from the right.

loc-ISL$^{(\sigma)}(l')$ ISLLL-reduces $R_{l'}^{(\sigma)}$. Initially, $H_{l'}^{(\sigma)}$ is the identity transform. $R_{l'}^{(\sigma)}$ and $H_{l'}^{(\sigma)}$ get updated for all reductions on level $\sigma-1$ done by **loc-ISL**$^{(\sigma-1)}(l)$. Upon termination the local transform $H_{l'}^{(\sigma)}$ done by **loc-ISL**$^{(\sigma)}(l')$ is transported to level $\sigma+1$.

We describe **loc-ISL**$^{(\sigma)}(l')$ more formally, first for $1 < \sigma < s-1$ thereafter for $\sigma = s-1$ and $\sigma = 1$.

loc-ISL$^{(\sigma)}(l')$ for $1 < \sigma < s-1$.
Given are $R_{l''}^{(\sigma+1)}$ for $l'' := \lceil l'/k_{\sigma+1} \rfloor$ and the transform $H_{l''}^{(\sigma+1)}$ that has been done locally on level $\sigma+1$ but not yet on level $\sigma+2$.

1. $l := k_\sigma(l'-1)+1$, (we always have that $l = \lceil l'/k_\sigma \rfloor$)
   form the submatrices $R_{l'}^{(\sigma)}$ of $R_{l''}^{(\sigma+1)}$ and $R_l^{(\sigma-1)}$ of $R_{l'}^{(\sigma)}$.
2. WHILE $l < k_\sigma(l'+1)$ DO
   loc-ISL$^{(\sigma-1)}(l)$ (ISLLL-reduces $R_l^{(\sigma-1)}$)
   IF $l > k_\sigma(l'-1)+1$ AND $D^{(\sigma-1)}(l-1) > (\alpha/\delta)^{(\mathbf{k}_{\sigma-1})^2} D^{(\sigma-1)}(l)$
   THEN $l := l-1$ ELSE $l := l+1$
3. Transport $H_{l'}^{(\sigma)}$ to level $\sigma+1$, reset $H_{l'}^{(\sigma)}$,
   Update $R_{l''}^{(\sigma+1)}$ into upper-triangular form.
4. Size-reduce $R_{l''}^{(\sigma+1)}$ and update $H_{l''}^{(\sigma+1)}$ accordingly.

Case $\sigma = s-1$. The global transforms of Steps 3, 4 are done on R and B. Steps 3 and 4 in **loc-ISL**$^{(s-1)}(l)$ are as follows:

3. Transform R and $B = [b_1, \ldots, b_n]$ globally by $H_{l'}^{(s-1)}$, reset $H_{l'}^{(s-1)}$.
   Update R into upper-triangular form.
4. Size-reduce $[B_{l''}^{(s)}, B_{l''+1}^{(s)}]$ globally and update R accordingly.

Case $\sigma = 1$. Steps 1-2 in $\textbf{loc-ISL}^{(1)}(l')$ are as follows:

1-2. First size-reduce then LLL-reduce $R^{(1)}_{l'}$.
After each reduction step update $R^{(1)}_{l'}$ into upper-triangular form.
Record the basis transform of $R^{(1)}_{l'}$ in the matrix $H^{(1)}_{l'}$.

Theorem 7. *Given a basis* $b_1, \dots, b_n \in \mathbf{Z}^d$, $n = k_1 \cdot \ldots \cdot k_s = \mathbf{k}_s$, *algorithm* **ISLLL** *performs at most* $O(dn^2 + d \log_{1/\delta} \overline{M} \sum_{\sigma=1}^{s} k_\sigma^2)$ *arithmetic steps and produces an ISLLL-reduced basis. If* $\max_\sigma k_\sigma = O(1)$ *and* $\log_2 \overline{M} = O(n^2)$, *the number of arithmetic steps is* $O(n^2 d\, s \log_{1/\delta} 2)$.

Proof. Correctness. **ISLLL** is correct since $\textbf{loc-ISL}^{(\sigma)}(l')$ locally ISLLL-reduces $[B^{(\sigma)}_{l'}, B^{(\sigma)}_{l'+1}]$ in coordinates of $R^{(\sigma)}_{l'}$, and transports the transform to level $\sigma+1$. A subsequent reduction of $R^{(\sigma)}_{l'\pm1}$ by $\textbf{loc-ISL}^{(\sigma)}(l' \pm 1)$ starts by copying $R^{(\sigma)}_{l'\pm1}$ from the updated $R^{(\sigma+1)}_{l''}$. $R^{(\sigma)}_{l'\pm1}$ are correct as both $R^{(\sigma)}_{l'-1}$ and $R^{(\sigma)}_{l'+1}$ are covered by $R^{(\sigma+1)}_{l''}$.

On termination all inequalities (1) of Definition 3 are satisfied and the resulting basis is an ISLLL-basis. Induction shows that whenever **ISLLL** calls $\textbf{loc-ISL}^{(\sigma)}(l')$ all inequalities (1) of Definition 3 are satisfied for the subbasis $b_1, ..., b_{\mathbf{k}_\sigma(l'-1)}$ that precedes segment $B^{(\sigma)}_{l'}$.

Time bound. Let $\mathrm{dec}^{(\sigma)}$ denote the number of times that $\textbf{loc-ISL}^{(\sigma)}(l'-1)$ is executed in **ISLLL** due to $D^{(\sigma)}(l'-1) > (\alpha/\delta)^{(\mathbf{k}_\sigma)^2} D^{(\sigma)}(l')$. The number of $\textbf{loc-ISL}^{(\sigma)}$ executions in **ISLLL** is $n/\mathbf{k}_\sigma - 1 + 2 \cdot \mathrm{dec}^{(\sigma)}$.

We apply Theorem 3 to segments and to the local Gramian determinants of level σ. Let

$$\mathbf{D}^{(\sigma)} =_{\mathrm{def}} \prod_{l=1}^{n/\mathbf{k}_\sigma} D_{\mathbf{k}_\sigma l} = \prod_{l=1}^{n/\mathbf{k}_\sigma} \left(D^{(\sigma)}(1) \cdot \ldots \cdot D^{(\sigma)}(l)\right).$$

Each execution of $\textbf{loc-ISL}^{(\sigma)}(l')$ — due to a violated inequality (1) — decreases $\mathbf{D}^{(\sigma)}$ by the factor $\delta^{(\mathbf{k}_\sigma)^2/2}$. Initially the integer $\mathbf{D}^{(\sigma)}$ satisfies $\mathbf{D}^{(\sigma)} \leq \overline{M}^{n/\mathbf{k}_\sigma}$, and upon termination $\mathbf{D}^{(\sigma)} \geq 1$, hence

$$\mathrm{dec}^{(\sigma)} \leq 2n(\mathbf{k}_\sigma)^{-3} \log_{1/\delta} \overline{M}.$$

In total there are $n/\mathbf{k}_\sigma - 1 + 2n(\mathbf{k}_\sigma)^{-3} \log_{1/\delta} \overline{M}$ executions of $\textbf{loc-ISL}^{(\sigma)}(l')$ each requiring an overhead of $O(\mathbf{k}_\sigma \mathbf{k}^2_{\sigma+1})$ arithmetic steps. This overhead covers: stepwise update of $R^{(\sigma)}_{l'}$, $H^{(\sigma)}_{l'}$ after each $\textbf{loc-ISL}^{(\sigma-1)}(l)$ execution, moreover final update of $R^{(\sigma+1)}_{l''}$ into upper-triangular form and size-reduction of $R^{(\sigma+1)}_{l''}$. The total overhead of all $\textbf{loc-ISL}^{(\sigma)}(l')$ executions is

$$O(n\mathbf{k}^2_{\sigma+1} + nk^2_{\sigma+1} \log_{1/\delta} \overline{M}).$$

In the particular case $\sigma = s-1$ the global transforms are done on the basis matrix $B \in \mathbf{Z}^{d\times n}$ and the overhead is

$$O(d\,\mathbf{k}_s^2 + d\,k_s^2 \log_{1/\delta} \overline{M}).$$

Moreover, $O(n \log_{1/\delta} \overline{M})$ local LLL-exchanges are done on level 1 in local coordinates of dimension $2k_1$ each requiring $O(k_1^2)$ local arithmetic steps.

We see that **ISLLL** performs $O(n^2 d + d \log_{1/\delta} \overline{M} \sum_{\sigma=1}^{s} k_\sigma^2)$ arithmetic steps. This proves the claimed time bound.

Size of integers. We extend the bounds on the size of integers in **SLLL**, shown in the proof Theorem 4, to **ISLLL**. The value $\max_i \|b_i\|$ can temporarily increase during final, global size-reduction at most by a factor $(\frac{3}{2})^{n-1}$.

Local size reduction is done in Step 4 of $\mathbf{loc\text{-}ISL}^{(\sigma)}(l')$ and in Steps 1-2 of $\mathbf{loc\text{-}ISL}^{(1)}(l)$. During size-reduction of b_i the preceding vectors $b_1, \dots, b_{i-1}$ must already be size-reduced. Then the argument of Theorem 4 applies and size-reduction of b_i can temporarily increase $\|b_i\|$ at most by a factor $(\frac{3}{2})^{i-1}$. Hence, $\|R_{l'}^{(\sigma)}\|_1 = 2^{O(n)}\overline{M}$ holds during local size reduction of $R_{l'}^{(\sigma)}$.

Next consider the size of the transform $H_{l'}^{(\sigma)}$ during $\mathbf{loc\text{-}ISL}^{(\sigma)}(l')$. $H_{l'}^{(\sigma)}$ transforms $[B_{l'}^{(\sigma)}, B_{l'+1}^{(\sigma)}]$ into segments that are locally ISLLL-reduced in the coordinates of $R_{l'}^{(\sigma)}$. The argument of Lemma 1 shows that $\|H_{l'}^{(\sigma)}\|_1 = 2^{O(\mathbf{k}_\sigma)}\overline{M}$.

Consider Step 3 of $\mathbf{loc\text{-}ISL}^{(\sigma)}(l')$ where $H_{l'}^{(\sigma)}$ is transported to level $\sigma + 1$. That transport means that submatrices of $R_{l''}^{(\sigma+1)}, H_{l''}^{(\sigma+1)}$ are multiplied by $H_{l'}^{(\sigma)}$. This matrix multiplication increases $\|R_{l''}^{(\sigma+1)}\|_1$, $\|H_{l''}^{(\sigma+1)}\|_1$ at most by a factor $2^{O(\mathbf{k}_\sigma)}\overline{M}$.

On termination of $\mathbf{loc\text{-}ISL}^{(\sigma)}(l')$ the final size-reduction of $R_{l''}^{(\sigma+1)}$ in Step 4 yields

$$\|R_{l''}^{(\sigma+1)}\|_1^2 \le \tfrac{\mathbf{k}_{\sigma+1}+1}{4}\overline{M} \ .$$

We see that all integers during excecution of **ISLLL** are bounded by $2^{O(n)}\overline{M}$. These integers are the numerators and denominators of the rational numbers $\mu_{j,i}, \|\widehat{b}_i\|^2$ and the integer coefficients of the basis vectors. If the input basis satisfies $\max_i \|b_i\| = 2^{O(n)}$ we have that $2^{O(n)}\overline{M} = 2^{O(n^2)}$ and **ISLLL** uses integers of bit length $O(n^2)$.

6 Strong ISLLL-Reduction

We strengthen ISLLL-bases of dimension $n = \mathbf{k}_s$ to strong ISLLL-bases (SISLLL-bases, for short) satisfying $\|b_1\|^2 \le \lambda_1^2 \, (\alpha/\delta^{s-1})^{n-1}$. This comes close to the property $\|b_1\|^2 \le \lambda_1^2 \, \alpha^{n-1}$ of LLL-bases. While LLL-bases are better than SISLLL-bases for the same δ, SISLLL-bases for $\delta' := \delta^{1/s}$ are better than LLL-bases for δ since $\alpha'/\delta'^{s-1} < \alpha$. Moreover, the **SISLLL**-time bound for δ' is at most s-times the time bound for δ.

Notation. We call segment $B_l = [b_{kl-k+1}, ..., b_{kl}]$ of lattice basis $b_1, \dots, b_n$, $k = 0 \mod n$, *σ-strong* if

$$\|b_1\|^2 / \|\widehat{b}_i\|^2 \le (\alpha/\delta^\sigma)^{i-1} \quad \text{for } i = kl - k + 1, \dots, kl$$

otherwise B_l is called *σ-weak.* B_l is called *completely σ-weak* if

$$\|b_1\|^2 / \|\widehat{b}_i\|^2 > (\alpha/\delta^\sigma)^{i-1} \quad \text{for } i = kl - k + 1, ..., k\, l.$$

We call the basis $b_1, \dots, b_n$ *σ-strong* if all segments are *σ-strong.* A basis is 1-strong if it is strong in the sense of Section 4. Every σ-strong basis satisfies

$$\|b_1\|^2 \,/\, \lambda_1^2 \;\leq\; (\alpha/\delta^\sigma)^{n-1},$$

since the argument of Lemma 2 applies to σ-strong bases.

Strong ISLLL (SISLLL)

INPUT ISLLL-basis $b_1, \dots, b_n \in \mathbf{Z}^d,\; n = k_1 \cdot \ldots \cdot k_s = \mathbf{k}_s$

OUTPUT $b_1, \dots, b_n$ $(s-1)$-strong ISLLL-basis

1. IF all segments $B_l^{(\sigma)}$ are $(s-1)$-strong
   THEN size-reduce $b_1, ..., b_n$ and terminate
2. **loc-ISL**$^{(\sigma)}(l)$ for the minimal σ and maximal l such that $B_{l+1}^{(\sigma)}$ is $(s-1)$-weak
3. WHILE $D^{(\sigma)}(l) > (\alpha/\delta^\sigma)^{(\mathbf{k}_\sigma)^2} D^{(\sigma)}(l+1)$ for some l, σ DO
   loc-ISL$^{(\sigma)}(l)$ for the smallest such σ
4. GO TO 1.

Algorithm **SISLLL** transforms an ISLLL-basis into an $(s-1)$-strong ISLLL-basis. Lemma 3 extends from **SSLLL** to **SISLLL** as follows.

Lemma 4. *If* **loc-ISL**$^{(\sigma)}(l)$ *is executed in* **SISLLL** *due to an $(s-1)$-weak $B_{l+1}^{(\sigma)}$ then the resulting $B_{l+1}^{(\sigma)}$ is $(s-1)$-strong or else the resulting $B_l^{(\sigma)}$ is completely $(s-1)$-weak, in which case we have that $D^{(\sigma)}(l-1) > (\alpha/\delta^{s-1})^{(\mathbf{k}_\sigma)^2} D^{(\sigma)}(l)$.*

Theorem 8. *The* **ISLLL**-*time bound of Theorem 7 also holds for* **SISLLL**.

Proof. By Lemma 4 a **loc-ISL**$^{(\sigma)}(l)$ execution in Step 2 of **SISLLL** either decreases

$$\mathrm{maw}^{(\sigma)} =_{\mathrm{def}} \max\{l \mid B_l^{(\sigma)} \text{ is } (s-1)-\text{weak}\},$$

or else there directly follows a **loc-ISL**$^{(\sigma)}(l-1)$ execution due to $D^{(\sigma)}(l-1) > (\alpha/\delta^{s-1})^{(\mathbf{k}_\sigma)^2} D^{(\sigma)}(l)$. The proof of Theorem 5 extends to the proof of Theorem 8. In particular, by Lemma 4 and Theorem 3 the number of **loc-ISL**$^{(\sigma)}(l)$ executions in **SISLLL** is at most $n/\mathbf{k}_\sigma + 4n(\mathbf{k}_\sigma)^{-3} \log_{1/\delta} M_{s_c}$. □

References

[BN00] D. Bleichenbacher and P.Q. Nguyen, *Noisy Polynomial Interpolation and Noisy Chinese Remaindering*, Eurocrypt 2000, Lecture Notes in Comput. Sci., 1807, Springer, New York, 2000, pp. 53-69.

[Bo00] D. Boneh, *Finding Smooth Integers in Small Intervals Using CRT Decoding*, ACM Symposium on the Theory of Computing 2000, ACM Press, 2000, pp. 265-272.

[Ca00] J. Cai, *The Complexity of some Lattice Problems*, Algorithmic Number Theory, Lecture Notes in Comput. Sci., 1838, Springer, New York, 2000, pp. 1-32.

[Co97] D. Coppersmith, *Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities*, *J. Cryptology*, 10, 1997, pp. 233-260.

[K84] R. Kannan, *Minkowski's Convex Body Theorem and Integer Programming*, Math. Oper. Res., 12, 1984, pp. 415–440.

[K01] H. Koy, Notes of a Lecture. Frankfurt 2001.

[KS01a] H. Koy and C.P. Schnorr, *Segment LLL-Reduction*, Cryptography and Lattices, Lecture Notes in Comput. Sci., 2146, Springer, New York, 2001, pp.67–80 (first version of the present paper).

[KS01b] H. Koy and C.P. Schnorr, *Segment LLL-Reduction with Floating Point Orthogonalization*, Cryptography and Lattices, Lecture Notes in Comput. Sci., 2146, Springer, New York, 2001, pp. 81–96.

[LLL82] A. K. Lenstra, H. W. Lenstra and L. Lovász, *Factoring polynomials with rational coefficients*, *Math. Ann.*, 261, 1982, pp. 515-534.

[NS00] P.Q. Nguyen and J. Stern, *Lattice Reduction in Cryptology, An Update*, Algorithmic Number Theory, Lecture Notes in Comput. Sci., 1838, Springer, New York, 2000, pp. 85-112.

[S87] C.P. Schnorr, *A hierarchy of polynomial time lattice basis reduction algorithms*, *Theoret. Comput. Sci.*, 53, 1987, pp. 201-224.

[S91] C.P. Schnorr and M. Euchner,*Lattice Basis Reduction and Solving Subset Sum Problems*, Fundamentals of Comput. Theory, Lecture Notes in Comput. Sci., 591, Springer, New York, 1991, pp. 68–85. The complete paper appeared in Math. Programming Studies, 66A, 2, 1994, pp. 181–199.

[S94] C.P. Schnorr, *Block Reduced Lattice Bases and Successive Minima*, *Combin. Probab. and Comput.* , 3, 1994, pp. 507-522.

[SH95] C.P. Schnorr and H. Hörner, *Attacking the Chor-Rivest Cryptosystem by Improved Lattice Reduction*, Eurocrypt 1995, Lecture Notes in Comput. Sci., 921, Springer, New York, 1995, pp. 1–12.

[Sc84] A. Schönhage, *Factorization of univariate integer polynomials by diophantine approximation and improved lattice basis reduction algorithm*, *Proc. 11-th Coll. Automata, Languages and Programming, Antwerpen 1984*, Lecture Notes in Comput. Sci., 172, Springer, New York, 1984, pp. 436-447.