Practical algorithms for constructing HKZ and Minkowski reduced bases

Wen Zhang, Sanzheng Qiao, and Yimin Wei

April 20, 2011

Abstract

In this paper, three practical lattice basis reduction algorithms are presented. The first algorithm constructs a Hermite, Korkine and Zolotareff (HKZ) reduced lattice basis, in which a unimodular transformation is used for basis expansion. Our complexity analysis shows that our algorithm is significantly more efficient than the existing HKZ reduction algorithms. The second algorithm computes a Minkowski reduced lattice basis. It is the first practical algorithm for Minkowski reduced bases for lattices of arbitrary dimensions. The third algorithm is an improvement of the second algorithm by drastically reducing the number of lattice points being searched. Since the original LLL algorithm is no longer applicable to the third algorithm, we propose a notion of quasi-LLL reduction to accelerate the computation.

Keywords Lattice, LLL reduced basis, HKZ reduced basis, Minkowski reduced basis, unimodular transformation.

1 Introduction

A *lattice* is a set of discrete points representing integer linear combinations of linearly independent vectors. The set of linearly independent vectors generating a lattice is called a *basis* for the lattice. A lattice basis is usually not unique, but all the bases have the same number of elements, called the *dimension* of the lattice. If the dimension of a lattice is larger than 1, there are infinitely many bases. For example, Figure 1 depicts the lattice generated by $\mathbf{a}_1 = [2.0, 0]^{\mathrm{T}}$ and $\mathbf{a}_2 = [2.7, 0.7]^{\mathrm{T}}$. This lattice has dimension 2, and for any integer c, $\mathbf{a}_1$ and $\mathbf{a}_2 + c \cdot \mathbf{a}_1$ form a basis for this lattice.

Since a lattice can have more than one basis, it is desirable to find one that is nearly orthogonal. It is reasonable to expect that the shorter the basis vectors are, the nearer they are to orthogonal. For example, $\mathbf{b}_1 = [-0.7, -0.7]^{\mathrm{T}}$ and $\mathbf{b}_2 = [1.3, -0.7]^{\mathrm{T}}$ form another basis for the lattice Figure 1. Figure 2 shows that the basis vectors $\mathbf{b}_1$ and $\mathbf{b}_2$ are shorter than $\mathbf{a}_1$ and $\mathbf{a}_2$ with respect to the L_2-norm, and they are nearer to orthogonal than $\mathbf{a}_1$ and $\mathbf{a}_2$.

A lattice basis consisting of relatively short vectors is called *reduced*. An ideally reduced basis consists of shortest possible vectors. The problem of finding good reduced bases is known as *lattice reduction*, which plays an important role in many fields of mathematics and computer science [3, 4, 13, 18, 21], particularly in communications [1, 23] and cryptology [6, 22]. There are vavious definitions of reduced bases. They differ in the degree of reduction.

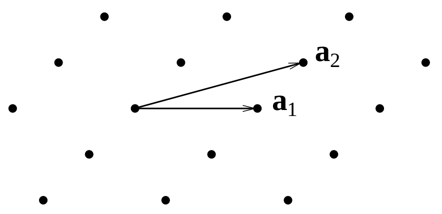

Figure 1: the lattice points generated by $\mathbf{a}_1$ and $\mathbf{a}_2$.

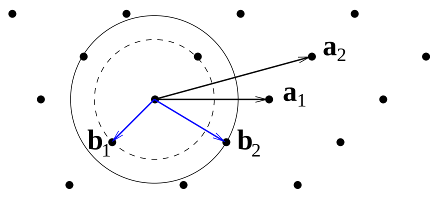

Figure 2: $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{b}_1$ $\mathbf{b}_2$ and the lattice.

In 1850, Hermite introduced the first notion of reduction for lattices of arbitrary dimensions, proposed an algorithm for computing such reduced bases, and proved its termination [25]. Hermite's algorithm is of theoretical significance, but its complexity is still unknown. Schnorr and Euchner [26] reconsidered this problem and developed a practical algorithm for constructing the Hermite reduced basis. In 1873, Korkine and Zolotareff [9] strengthened the definition of Hermite reduced basis. Their proposed notion of reduction is usually called the *HKZ reduced basis* [13], named after Hermite, Korkine and Zolotareff. In 1983, using induction, Kannan [7] presented the first algorithm for constructing the HKZ reduced bases. Helfrich [17], Kannan [8], and Banihashemi and Khandani [27] further refined Kannan's algorithm and improved the complexity analysis. Note that the methods based on Kannan's strategy are intended as theoretical tools, and the related papers usually focus on asymptotic complexity. Agrell et. al. [1] presented a practical algorithm and used it as a preprocessor for the integer least squares problems.

In 1891, Minkowski [14] defined a new notion of reduction, which is stronger than the HKZ reduction. This definition is now known as the *Minkowski reduced basis.* The Minkowski reduced bases are of fundamental importance in many fields of mathematics. For example, they are used in assessing the quality of random number generators [18] and in the reduction of quadratic forms in number theory [3].

The construction of Minkowski reduced bases is a classical problem which attracts much attentions. Marsaglia [19] and Lagarias [15] presented two algorithms for constructing Minkowski reduced bases for lattices of dimensions 2 and 3, respectively. Beyer, Roof, and Williamson [20] presented a practical algorithm for lattices of dimensions less than 7. Algorithms for constructing the Minkowski reduced bases for lattices of arbitrary dimensions can be found in [16, 17]. However, these algorithms are impractical for high dimensional lattices since there are stringent restrictions on using them.

Both the construction of the HKZ reduced bases and the construction of the Minkowski reduced bases consist of a sequence of the *shortest point search* problems. The shortest point search problem, which is actually a special integer least squares problem, is to find a shortest lattice point with respect to the L_2-norm in a given lattice. This problem has been proven to be NP-hard [28]. Even finding an approximate solution with which the ratio between the computed distance and the shortest distance is upper-bounded by a constant, is also NP-hard [29]. Therefore, the construction of an HKZ reduced or a Minkowski reduced basis requires intensive computation. This motivates Lenstra, Lenstra, and Lovász [10] to develop the first polynomial-time lattice reduction algorithm, known as the LLL algorithm, named after the three authors. Their notion of reduced basis is actually a relaxation of the Hermite reduced basis [25]. The LLL algorithm has become the most important tool in public-key cryptanalysis [24] and integer least squares problems [1, 30]. Further improvements of the LLL algorithm have been developed. While some [31, 11, 34] improve the quality of the output of the LLL algorithm, others [32, 23] improve the efficiency of the algorithm.

In this paper, we present three practical algorithms: one for constructing the HKZ reduced bases and two for constructing the Minkowski reduced bases. The first algorithm uses the same shortest point search algorithm during the recursive process as the algorithm in [1]. However, it uses a different method for the expansion of a shortest vector into a new lattice basis. In [1], the basis expansion strategy introduced by Kannan [7] is used, while in our new algorithm, the unimodular transformation technique presented in [33] is used. Note that the Kannan's expansion strategy only works for rational lattices, while the unimodular transformation technique works for any real lattice and is much more efficient than the Kannan's strategy.

The other two algorithms presented in this paper are focused on the construction of a Minkowski reduced basis for a given lattice. In general, both algorithms are based on the Schnorr and Euchner's search strategy [26] and the unimodular transformation technique [33]. Specifically, the first algorithm uses a simple variation of the shortest point search algorithm presented in [1] to compute a Minkowski reduced basis vector, while the second algorithm dynamically monitors the basis expansion condition during the search process. Thus, the second algorithm is more efficient than the first one. However, the first algorithm can be preconditioned by using the LLL algorithm, while the second one can not. In order to accelerate the second algorithm, we propose a new lattice reduction method, called "incomplete LLL algorithm", as a preprocess for the second algorithm. Numerical results show that the combination of the second algorithm and the incomplete LLL algorithm is much faster than the combination of the first algorithm and the LLL algorithm, and both of them significantly outperform the existing algorithms for the Minkowski reduced bases.

The rest of the paper is organized as follows. In Section 2, we review several concepts on lattices and bases. In Sections 3 and 4, we introduce the famous LLL and sphere decoding algorithms which will be used later. The new algorithm for constructing an HKZ reduced basis is given in Section 5. Section 6 presents the first algorithm for constructing a Minkowski reduced basis. The incomplete LLL algorithm and the second algorithm for constructing a Minkowski reduced basis are presented in Section 7. Finally, the paper is concluded in Section 8.

2 Preliminaries

In this section, we briefly review some basic concepts in the field of lattice theory. Given a real matrix $B \in \mathbb{R}^{m \times n}$, $m \geq n$, of full column rank, the set

$$L = \{B\mathbf{z}, \quad \text{for all integer } n\text{-vectors } \mathbf{z} \in \mathbb{Z}^n\},$$

containing discrete grid points, is called a *lattice* generated by B. The linearly independent columns of B form a *basis* for L.

A set of lattice points does not uniquely determine a basis, but all the bases have the same number n of elements, called the *dimension* of L. In general, if $Z \in \mathbb{R}^{n\times n}$ is an integer matrix whose inverse is also an integer matrix, then both B and BZ generate the same lattice. An integer matrix $M \in \mathbb{R}^{n\times n}$ is called *unimodular* if $\det(M) = \pm 1$, where $\det(M)$ denotes the determinant of M. It is obvious that an integer nonsingular matrix Z is unimodular if and only if its inverse is also integer. Thus two different bases of a lattice are related with each other by a unimodular matrix.

The *determinant* of a lattice is defined by

$$\det(L) = \sqrt{\det(B^{\mathrm{T}}B)},$$

where B is a generating matrix of the lattice L. From the definition of unimodular, $\det(B^{\mathrm{T}}B) = \det((BZ)^{\mathrm{T}}BZ)$, for any unimodular Z. Hence the determinant of a lattice is independent of the choice of basis. Actually, the determinant of a lattice can be viewed as the volume of the parallelepiped spanned by a basis for the lattice.

Since a lattice can have many bases, it is desirable to find one that is nearly orthogonal. The problem for finding good reduced bases is known as *lattice reduction*. In terms of matrix, the goal of lattice reduction is to find a unimodular Z so that the columns of BZ form a reduced basis.

There are various definitions of a reduced basis, and most of them are based on the orthogonal decomposition of the lattice generating matrix that can be achieved by the following procedure.

Procedure 1 (Gram-Schmidt Orthogonalization) *Given a matrix $B = [\mathbf{b}_1, \cdots, \mathbf{b}_n] \in \mathbb{R}^{m\times n}$, this procedure produces matrices Q^* and U such that $B = Q^*U$, where $Q^* \in \mathbb{R}^{m\times n}$ has orthogonal columns and $U \in \mathbb{R}^{n\times n}$ is upper triangular with a unit diagonal.*

```
1. $U \leftarrow n \times n$ identity matrix $I_n$.
2. $\mathbf{q}_1^* \leftarrow \mathbf{b}_1$.
3. for $j = 2 : n$
4.     for $i = 1 : j-1$
5.         $U(i,j) \leftarrow \frac{(\mathbf{b}_j, \mathbf{q}_i^*)}{\|\mathbf{q}_i^*\|_2^2}$.
6.     end
7.     $\mathbf{q}_j^* \leftarrow \mathbf{b}_j - \sum_{i=1}^{j-1} U(i,j)\mathbf{q}_i^*$.
8. end
9. $Q^* \leftarrow [\mathbf{q}_1^*, \cdots, \mathbf{q}_n^*]$.
```

Note that the above Gram-Schmidt orthogonalization (GSO) procedure is square-root free. So the GSO can be performed in exact arithmetic for integer or rational lattice generating matrices, since all the operations are rational arithmetic. Therefore the GSO is widely used in applications of cryptology, in which the lattice generating matrices usually appear to be integral [24]. However, the generating matrices in the applications of communications are usually real or complex [1, 23], and the computations of the GSO are not numerically stable for such matrices [5]. So in such applications, the GSO is usually replaced by a mathematically equivalent but numerically stable QR decomposition, obtained by applying a sequence of Householder or Givens transformations [5, 11]:

$$B = QR, \tag{1}$$

where Q has orthonormal columns and $R = [r_{i,j}]$ is an upper triangular matrix with positive diagonal. Setting the diagonal matrix $D = \text{diag}(d_i)$ with $d_i = \|\mathbf{q}_i^*\|_2$, then the GSO and the QR decomposition are related by

$$Q^* = QD \quad \text{and} \quad R = DU.$$

For simplicity, in this paper, we consider floating-point lattice basis matrices arised from communications and use the QR decomposition approach (1). It is straightforward to convert our derivations into the integer case.

To reduce the lengths of basis vectors, Hermite firstly introduced a weak notion of reduction [13, Page 37] that is often a necessary condition for a reduced basis.

Definition 1 (Size-reduced) *A lattice basis $\{\mathbf{b}_1, \cdots, \mathbf{b}_n\} \subseteq \mathbb{R}^m$ is called size-reduced if the upper triangular factor R in the QR decomposition (1) of $B = [\mathbf{b}_1 \cdots, \mathbf{b}_n]$ satisfies*

$$|r_{i,j}| \leq \frac{1}{2}|r_{i,i}|, \quad \textit{for } 1 \leq i < j \leq n.$$

Let $\lfloor a \rceil$ denote an integer closest to a, then a size-reduced basis of a given lattice can be obtained by the following process.

Algorithm 1 (Size-reduction) *Given a lattice basis $\{\mathbf{b}_1, \cdots, \mathbf{b}_n\} \subseteq \mathbb{R}^m$, this algorithm computes matrices Q with orthonormal columns, R upper triangular, and Z unimodular such that $BZ = QR$ is size-reduced.*

```
1.  Initial QR decomposition: $B = QR$.
2.  Initial $Z = I_n$;
3.  for $j = 2 : n$
4.      for $i = j - 1 : -1 : 1$
5.          Reduce$(i, j)$;
6.      endfor
7.  endfor
```

The procedure `Reduce`(i, j) on line 5 reduces the size of $r_{i,j}$.

Procedure 2 (`Reduce`(i, j)) *Given R and Z, this procedure updates R and Z such that the updated R satisfies $|r_{i,i}| \geq 2\,|r_{i,j}|$.*

```
1.  Calculate $\gamma = \lfloor r_{i,j}/r_{i,i} \rceil$;
2.  if $\gamma \neq 0$
3.      Form $Z_{ij} = I_n - \gamma \mathbf{e}_i \mathbf{e}_j$, where $\mathbf{e}_i$ is the $i$th unit vector;
4.      Apply $Z_{ij}$ to $R$ and $Z$: $R \leftarrow RZ_{ij}$ and $Z \leftarrow ZZ_{ij}$;
5.  endif
```

In 1850, Hermite [25] proposed a notion of reduction for arbitrary dimensional lattices. Korkine and Zolotareff [9] further strengthened the definition and proposed a new notion called *HKZ reduced basis* [13], named after Hermite, Korkine and Zolotareff.

Definition 2 (HKZ reduced) *A lattice basis $\{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_n\} \subseteq \mathbb{R}^m$ is called HKZ reduced if it is size-reduced, and for each trailing $(n-i+1)$-by-$(n-i+1)$, $1 \leq i \leq n$, submatrix of the upper triangular matrix R in the decomposition (1) of $B = [\mathbf{b}_1\ \mathbf{b}_2\ ...\ \mathbf{b}_n]$, its first column is a shortest nonzero vector in the lattice generated by the submatrix.*

From Definition 2, an HKZ reduced basis contains a shortest nonzero lattice vector with respect to the L_2-norm as its first element.

Minkowski [14] in 1891 defined a new notion of reduction which is stricter than the HKZ reduced basis, known as the *Minkowski reduced basis.*

Definition 3 (Minkowski reduced) *A lattice basis $\{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_n\} \subseteq \mathbb{R}^m$ is called Minkowski reduced if for $i = 1, \cdots, n$, $\mathbf{b}_i$ is a shortest lattice vector so that $\{\mathbf{b}_1, ..., \mathbf{b}_i\}$ can be extended to a basis for L.*

It can be seen from Definition 3 that Minkowski reduced bases require each element as short as possible, and this concept can be considered as the strongest definition of a reduced basis. Of course, as an HKZ reduced basis, a Minkowski reduced basis contains a shortest nonzero lattice point as its first element.

3 The LLL Reduction

Although the reduction quality of an HKZ reduced or a Minkowski reduced bases is good, the corresponding algorithms for obtaining them are computationally intensive, since both of the two reduction notions are associated with the shortest point search problem, which has been proven to be NP-hard [28].

Lenstra, Lenstra, and Lovász [10] reconsidered the Hermite's algorithm [25] and proposed a relaxed version of the HKZ reduced basis as follows:

Definition 4 (LLL reduced) *Given an $\omega \in (0.25, 1.0)$, a lattice basis $\{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_n\}$ is called LLL-reduced if the upper triangular matrix R in the decomposition (1) of $B = [\mathbf{b}_1\ \mathbf{b}_2\ ...\ \mathbf{b}_n]$ satisfies*

$$|r_{i,j}| \leq |r_{i,i}|/2, \quad 1 \leq i < j \leq n \quad \textit{(size-reduced)}, \tag{2}$$

and

$$r_{i,i}^2 + r_{i-1,i}^2 \geq \omega\, r_{i-1,i-1}^2, \quad 1 < i \leq n. \tag{3}$$

It is easy to see that both an HKZ reduced basis and an LLL reduced basis require size-reduction. The main difference between them is that for each trailing $(n-i+1)$-by-$(n-i+1)$ submatrix of the upper triangular matrix R in the decomposition (1) of the lattice generating matrix, an HKZ reduced basis requires that its first column be a shortest nonzero vector in the lattice generated by the submatrix, while an LLL reduced basis only requires that its first column scaled by a factor of ω be shorter than its second column. Thus, an HKZ reduced basis is LLL reduced for any $0.25 < \omega < 1$. To justify that an LLL reduced basis consists of vectors reasonably short, it is shown in [10] that if $\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_n$ form an LLL-reduced basis for a lattice, then

$$\eta^{1-i}\lambda_i^2 \le \|\mathbf{b}_i\|_2^2 \le \eta^{n-i}\lambda_i^2, \tag{4}$$

where $\eta = (\omega - 1/4)^{-1}$ and $\lambda_1, \lambda_2, ..., \lambda_n$ are the Minkowski minima [3, Page 201]. In particular, when $\omega = 3/4$, then $\eta = 2$ and

$$2^{1-i}\lambda_i^2 \le \|\mathbf{b}_i\|_2^2 \le 2^{n-i}\lambda_i^2.$$

In the LLL algorithm, the condition (2) is enforced by Procedure 2, the condition (3) is enforced by the following procedure, which swaps columns $i-1$ and i of R and restores its upper triangular structure.

Procedure 3 **(`SwapRestore`(i))** *Given matrices Q with orthonormal columns, R upper triangular, and Z unimodular, this procedure updates Q, R, and Z such that the update R satisfies the condition (3).*

```
1.  Compute the plane reflection $G$ such that
    $G \begin{bmatrix} r_{i-1,i-1} & r_{i-1,i} \\ 0 & r_{i,i} \end{bmatrix} P$ is upper triangular, where $P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$;
2.  Apply $Q_i = \text{diag}([I_{i-2}\ G\ I_{n-i}])$ and permutation $\Pi_i = \text{diag}([I_{i-2}\ P\ I_{n-i}])$ to
    $Q$, $R$, and $Z$: $Q \leftarrow QQ_i$, $R \leftarrow Q_i R \Pi_i$, $Z \leftarrow Z\Pi_i$;
```

The famous LLL algorithm, which is widely used in many fields of cryptology [24] and communications [1, 30] due to its high efficiency, is presented in terms of matrix [11] as follows:

Algorithm 2 **(`LLL`(B, ω))** *Given a lattice generating matrix $B = [\mathbf{b}_1, \cdots, \mathbf{b}_n] \in \mathbb{R}^{m\times n}$ and a parameter ω, $0.25 < \omega < 1$, this algorithm computes a unimodular matrix $Z \in \mathbb{Z}^{n\times n}$ and the upper triangular matrix factor $R \in \mathbb{R}^{n\times n}$ in the QR decomposition of BZ, so that R satisfies the conditions (2) and (3) in Definition 4. In other words, columns of BZ form an LLL reduced basis for the lattice.*

```
1.  Compute the QR decomposition: $B = QR$;
2.  Set $Z \leftarrow I$;
3.  $k \leftarrow 2$;
4.  while $k \le n$
5.      Reduce($k-1, k$);
6.      if $r_{k,k}^2 + r_{k-1,k}^2 < \omega\, r_{k-1,k-1}^2$;
7.          SwapRestore($k$);
8.          $k \leftarrow \max(k-1, 2)$;
9.      else
10.         for $i = k-2$ downto 1
11.             Reduce($i, k$);
12.         endfor
13.         $k \leftarrow k+1$;
14.     endif
15. endwhile
```

It is shown in [10] that the LLL algorithm has polynomial-time complexity for integer lattices. For real or complex lattices, numerical results [11, 23, 30] have shown that the LLL algorithm is also very efficient, although its complexity in such cases is still unknown.

The LLL algorithm has now become the most practical tool in many fields in mathematics, computer science, and communications. For example, in wireless multiple-antennas systems, the LLL algorithm is employed to improve the performance of multiinput multioutput (MIMO) detection [23, 35, 36]

4 The Closest Point Search Problem

As pointed out previously, the calculation of the HKZ or the Minkowski reduced bases involves solving a sequence of shortest point search problems. The shortest point search problem is closely related to the *closest point search* problem. Let $B \in \mathbb{R}^{m\times n}$ and let $\mathbf{x} \in \mathbb{R}^m$ be an observed vector, then the closet point search problem is actually the integer least squares problem:

$$\min_{z}\{\|Bz - \mathbf{x}\|_2 : \ \mathbf{z} \in \mathbb{Z}^n\} \tag{5}$$

The optimization problem (5) is to find a lattice point Bz that is closest the observed vector $\mathbf{x}$. In particular, when $\mathbf{x} = \mathbf{0}$, then the closest point search problem reduces to the shortest point search problem.

The closest point search problem arises in many applications in communications. In channel coding, if a lattice is used as a code for Gaussian channel, the maximum-likelihood estimation leads to a closest point search problem of the form (5), and the process for solving this problem is usually referred to as *decoding* [1, 35, 36].

The choice of method for solving the closest point problem (5) depends on the structure of the lattice generating matrix. For many classical lattices, efficient search algorithms exploiting the special structure of the lattice generating matrix are known [37, 38]. However, since the optimization problem (5) is proven to be NP-hard [28], the cost of theses algorithms quickly becomes prohibitive as the dimension of the lattice increases.

For the general lattice closest point search problem, that is, the lattice generating matrix has no exploitable structure, there are mainly two strategies: Kannan's strategy [7] and Pohst's strategy [40]. The two strategies are unified in the same framework in [1]. In general, the common feature of most algorithms for solving this problem is to first identify a region in which the optimal solution of (5) must lie, and then exhaustively search the lattice points in this region for a closest point, while possibly reducing the size of the region dynamically.

Specifically, let $B \in \mathbb{R}^{m\times n}$ is of full column rank and let $L(B)$ be the lattice generated by B. We write B as

$$B = [B_1, \mathbf{b}_n],$$

where B_1 is an $m \times (n-1)$ matrix consisting of the first $n-1$ columns of B and $\mathbf{b}_n$ is the last column of B. Then $L(B)$ can be decomposed into a stack of $(n-1)$-dimensional sublattices:

$$L(B) = \bigcup_{u_n=-\infty}^{+\infty} \{\mathbf{c} + u_n\mathbf{b_n} : \ \mathbf{c} \in L(B_1)\}, \quad u_n \in \mathbb{Z}. \tag{6}$$

It is easy to see that these sublattices distinguish from each other by the shift $u_n\mathbf{b}_n$, thus u_n is called the *index* of the sublattice. The index indicates the sublattice in which a certain lattice point lies. Denote $\mathbf{b}_\perp$ the orthogonal projection of $\mathbf{b}_n$ on the orthogonal complementary subspace $\mathcal{R}(B_1)^\perp$ of the range space $\mathcal{R}(B_1)$ of B_1. Then a sublattice with index u_n is contained in a subspace S_{u_n} defined by

$$S_{u_n} = \{\mathbf{v} + u_n \mathbf{b}_\perp : \ \mathbf{v} \in \mathcal{R}(B_1)\} \tag{7}$$

Let $\mathbf{x} \in \mathbb{R}^m$ be the observed vector to be decoded in the lattice $L(B)$. The orthogonal distance from $\mathbf{x}$ to the subspace S_{u_n} is

$$d(\mathbf{x}, S_{u_n}) = |u_n - \hat{u}_n| \cdot \|\mathbf{b}_\perp\|_2, \quad \text{where} \quad \hat{u}_n = \frac{\mathbf{x}^{\mathrm{T}}\mathbf{b}_\perp}{\|\mathbf{b}_\perp\|_2^2}. \tag{8}$$

Let $\hat{\mathbf{x}}$ denote the closest lattice point to $\mathbf{x}$, and suppose that $\hat{\mathbf{x}}$ lies in a sublattice S_t defined in (6) with index $t \in \mathbb{Z}$. If an upper bound ρ_n on $\|\hat{\mathbf{x}} - \mathbf{x}\|_2$ is given, then

$$d(\mathbf{x}, S_t) \le \rho_n. \tag{9}$$

From (8) and (9), $\hat{\mathbf{x}}$ must lie in one of the sublattices indexed by

$$u_n = \left\lceil \hat{u}_n - \frac{\rho_n}{\|\mathbf{b}_\perp\|_2} \right\rceil, \ldots, \left\lfloor \hat{u}_n + \frac{\rho_n}{\|\mathbf{b}_\perp\|_2} \right\rfloor, \tag{10}$$

a finite sequence of consecutive integers. Therefore, an n-dimensional closest point search problem can be reduced to a finite number of $(n-1)$-dimensional closest point search problems, leading to a recursive method for solving the problem (5). In summary, we present the following pseudo code for solving the closest point search problem.

Algorithm 3 (`Decode`(B,$\mathbf{x}$)) *Given a matrix $B \in \mathbb{R}^{m \times n}$ ($m \ge n$) of full column rank and a vector $\mathbf{x} \in \mathbb{R}^m$, this algorithm computes a solution of the closest point search problem (5).*

```
1. if $m = 1$
2.       return $z = \lfloor \frac{\mathbf{x}}{B} \rceil$;
3. else
4.       Determine an initial size $\rho_n$ of the search region;
5.       Compute $\mathbf{b}_\perp$ defined previously and $\hat{u}_n$ in (8);
6.       For each index $u_n$ in (10), solve the $(n-1)$-dimensional
         closest point search problem
                   $\mathbf{w}_{u_n} = \texttt{Decode}(B_1, \mathbf{x} - \hat{u}_n \mathbf{b}_\perp - u_n \mathbf{b})$
         in a search region with an updated size $\rho_{n-1}$;
7.       For each index $u_n$ in (10), construct $\mathbf{z}_{u_n} = [\mathbf{w}_{u_n}\ u_n]^{\mathrm{T}}$, find $\mathbf{z}$
         minimizing $\|B\mathbf{z}_{u_n} - \mathbf{x}\|_2$, and return $\mathbf{z}$.
```

There are various implementations of Algorithm 3. The main differences among them have the following three aspects:

- The choice of the initial size ρ_n of search region. The most algorithms based on the Kannan strategy [7, 17, 8, 27] or the Phost strategy [39, 40, 41] take the distance between $\mathbf{x}$ and the first column of the lattice generating matrix as the initial size. A better choice is the *Babai* *nearest point* [42], which is usually closer to $\mathbf{x}$ than the first column of the lattice generating matrix. The most algorithms based on the Schnorr-Euchner strategy [26, 1] take the distance between $\mathbf{x}$ and the Babai nearest point as the initial size. Furthermore, algorithms based on the Phost or the Schnorr-Euchner strategy reduce the size ρ_n dynamically to improve their performance. That is, when any lattice point $\mathbf{x}'$ inside the search region is found, the bound ρ_n can be reduced to $\|\mathbf{x}' - \mathbf{x}\|_2$, since $\|\mathbf{x}' - \mathbf{x}\|_2 \leq \rho_n$. The algorithms based on the Kannan strategy scan all the $(n-1)$-dimensional sublattices with the same value of ρ_n.

- The choice of the updated upper bound ρ_{n-1}. Generally, the algorithms based on the Kannan strategy search all the sublattices with indices in (10) using the same value of ρ_{n-1}. The methods in [7], [17], [8], and [27] differ from each other mainly on how the updated bounds ρ_k, $k = 1, ..., n$, are chosen. The algorithms based on the Phost [39, 40, 41] or the Schnorr-Euchner [26, 1] strategy update $\rho_{n-1} = \sqrt{\rho_n^2 - d(\mathbf{x}, S_{u_n})^2}$, which differs from sublattice to sublattice. Geometrically, the lattice points inside a hypersphere are searched.

- The order in which the sublattices are examined. The algorithms based on the Kannan or the Phost strategy scan all the $(n-1)$-dimensional sublattices with indices in (10) following a natural order. Assuming that $\hat{u}_n \leq \lfloor \hat{u}_n \rceil$, the algorithms based on the Schnorr-Euchner strategy [26, 1] search sublattices with indices following the alternating order:

$$u_n = \lfloor \hat{u}_n \rceil, \lfloor \hat{u}_n \rceil - 1, \lfloor \hat{u}_n \rceil + 1, \lfloor \hat{u}_n \rceil - 2, \ldots. \tag{11}$$

  The order in (11) is obtained according to nondecreasing distances $d(\mathbf{x}, S_{u_n})$. By using this search order, the chance of finding the correct sublattice early is maximized. So the algorithms based on the Schnorr-Euchner strategy are usually more efficient than those based on the Kannan or the Phost stategy.

In addition to the above three aspects, the structure of the given lattice generating matrix also has a significant impact on the efficiency of the decoding algorithm. All the algorithms mentioned above usually use the LLL algorithm as a preprocessor, since the performance of the decoding algorithm can be further improved for the LLL reduced bases. For applications where the same lattice is searched many times, a better choice is to use the HKZ reduction algorithm as a preprocessor [1].

5 A New Algorithm for Constructing the HKZ Reduced Bases

In 1983, Kannan [7] proposed the first algorithm for constructing the HKZ reduced bases for general lattices. Based on Kannan strategy, Helfrich [17], Kannan [8], and Banihashemi and Khandani [27] further refined Kannan's algorithm. All the algorithms based on the Kannan strategy are intended as theoretical results rather than practical tools, since the induction conditions imposed by the Kannan strategy are crucial and the complexity quickly becomes prohibitive as the dimension of lattices increases.

From Definition 2, the key to the construction of an HKZ reduced basis is to recursively find a shortest nonzero lattice vector and then to extend this vector to a basis for the lattice. From Section 4, the Schnorr-Euchner strategy is currently the most efficient method for finding a shortest nonzero lattice point. Agrell et al. [1] presented a practical implementation of the Schnorr-Euchner strategy and combine their sphere decoding algorithm with the Kannan's basis expansion algorithm to construct an HKZ reduced basis for a given lattice.

In this section, we present a new algorithm for computing the HKZ reduced bases for general lattices. In our algorithm, we adopt the sphere decoding method in [1] to compute a shortest nonzero lattice vector. However, instead of the Kannan's basis expansion method used in [1], we use a novel unimodular transformation basis expansion strategy.

Firstly, we state the Kannan's basis expansion method.

Algorithm 4 (SELECT-BASIS(n; $\mathbf{b}_1, \cdots, \mathbf{b}_{n+1}$)[8]) *Suppose that $\mathbf{b}_1, \cdots, \mathbf{b}_{n+1}$ are vectors in $\mathbb{Q}^m$ for some $m \geq n$ and span an n-dimensional lattice L of $\mathbb{Q}^m$. This procedure returns a basis $\{\mathbf{a}_1, \cdots, \mathbf{a}_n\}$ for L, where $\mathbf{a}_1$ is a shortest lattice vector in the direction of $\mathbf{b}_1$.*

```
1.  if $n = 0$ or $\mathbf{b}_1 = 0$ then do the obvious;
2.  if $\mathbf{b}_1$ is independent of $\mathbf{b}_2, \cdots, \mathbf{b}_{n+1}$
3.      $\mathbf{a}_1 \leftarrow \mathbf{b}_1$;
4.  else
5.      Find the rationals $\alpha_2, \cdots, \alpha_{n+1}$ (unique) such that $\sum_{j=2}^{n+1} \alpha_j \mathbf{b}_j = \mathbf{b}_1$;
6.      $M \leftarrow$ least common multiples of the denominators of $\alpha_2, \cdots, \alpha_{n+1}$;
7.      $\gamma \leftarrow \gcd(M\alpha_2, \cdots, M\alpha_{n+1})$;
8.      Let $M/\gamma = p/q$, where $p, q$ are relatively prime integers. $\mathbf{a}_1 \leftarrow (1/q)\mathbf{b}_1$;
9.  endif
10. $\overline{\mathbf{b}}_i \leftarrow \mathbf{b}_i - \frac{\langle \mathbf{b}_i, \mathbf{a}_1 \rangle}{\langle \mathbf{a}_1, \mathbf{a}_1 \rangle} \mathbf{a}_1, \quad i = 2, \cdots, n+1$;
11. $(\mathbf{c}_2, \cdots, \mathbf{c}_n) \leftarrow$ SELECT-BASIS($n-1$; $\overline{\mathbf{b}}_2, \cdots, \overline{\mathbf{b}}_{n+1}$);
12. Lift $\mathbf{c}_i$ to $\mathbf{a}_i$ in $L$ for $i = 2, \cdots, n$ and return $(\mathbf{a}_1, \cdots, \mathbf{a}_n)$.
```

Secondly, we give a brief complexity analysis of Algorithm 4. Step 11 shows that the algorithm is recursive and each time the problem size is reduced by one, from n downto 1. In each recursion, the major cost is in steps 2 and 5, which involve solving systems of linear equations requiring $O(k^3)$ arithmetic operations, assuming the k is the problem size in the recursion. Thus the complexity of Algorithm 4 is $O(n^4)$ ($O(\sum_{k=1}^{n} k^3)$). Moreover, note that Algorithm 4 only works for rational lattices, not general real lattices.

We propose a basis expansion method based on the unimodular transformation presented in [33], which is applicable for general real or complex lattices. Specifically, let $B \in \mathbb{R}^{m \times n}$ be a matrix generating the lattice L. Suppose that $B\mathbf{z}$ is a shortest nonzero point of L, where $\mathbf{z} = [z_i] \in \mathbb{Z}^n$. Then the problem of expanding $B\mathbf{z}$ to a basis of L is equivalent to the problem of constructing an n-by-n unimodular matrix Z whose first column is z, because the columns of BZ form a basis for L. In other words, $Z^{-1}\mathbf{z} = \mathbf{e}_1$, which says that Z^{-1}, also unimodular, transforms $\mathbf{z}$ into the first unit vector $\mathbf{e}_1$.

For the special case when $n = 2$, we have the following algorithm:

Procedure 4 (Unim2(p, q)[33]) *Let $[p \;\; q]^{\mathrm{T}}$ be a nonzero integer vector and $\gcd(p, q) = d$. Using the extended Euclidean algorithm, find integers a and b such that $ap + bq = d$. The integer matrix*

$$M = \begin{bmatrix} p/d & -b \\ q/d & a \end{bmatrix}, \tag{12}$$

is unimodular and

$$M^{-1}\begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} d \\ 0 \end{bmatrix}, \quad M^{-1} = \begin{bmatrix} a & b \\ -q/d & p/d \end{bmatrix}.$$

The above procedure shows that given a nonzero integer vector $[p \;\; q]^{\mathrm{T}}$, $\gcd(p,q) = d$, we can construct an integer unimodular matrix (12) whose inverse can be applied to $[p \;\; q]^{\mathrm{T}}$ to annihilate its second entry. In particular, if $\gcd(p,q) = \pm 1$, then $[p \;\; q]^{\mathrm{T}}$ can be transformed into the first unit vector.

Now we consider the general case when $n > 2$. Since $B\mathbf{z}$ is a shortest nonzero lattice point, we have $\gcd(z_i) = \pm 1$, implying that a sequence of the plane unimodular transformations M described above can be applied to transform $\mathbf{z}$ into the first unit vector.

As pointed out earlier, the construction of an HKZ reduced basis involves the shortest lattice vector search problem and we adopt the sphere decoding method. Since the LLL algorithm introduced in Section 3 can significantly accelerate the search process of the sphere decoding algorithm [1], we use the LLL algorithm as a preprocessor.

Putting all things together, we present our new algorithm for constructing an HKZ reduced basis.

Algorithm 5 (`LLL-aid-HKZ-red`(B, w)) *Given a lattice generator matrix $B \in \mathbb{R}^{m\times n}$ and a parameter ω, where $0.25 < \omega < 1$, this algorithm computes a unimodular matrix $Z \in \mathbb{Z}^{n\times n}$ such that the columns of BZ form an HKZ reduced basis.*

```
1.  Compute the QR decomposition: $B = QR$;
2.  $Z \leftarrow I_n$;
3.  for $k = 1$ to $n-1$
4.      Apply the LLL algorithm to $R(k:n, k:n)$;
5.      Use the sphere decoding algorithm [1] to find a nonzero vector $\mathbf{z} \in \mathbb{Z}^{n-k+1}$ so that
        $R(k:n, k:n)\mathbf{z}$ is a shortest point in the lattice generated by $R(k:n, k:n)$;
6.      Transform$(k, \mathbf{z})$;
7.  endfor
8.  for $j = 2$ to $n$
9.      for $i = j-1$ downto 1
10.         Reduce$(i, j)$;
11.     endfor
12. endfor
```

The procedure `Transform` in line 6 expands the shortest nonzero lattice vector found in line 5 to a basis for the $n-k+1$-dimensional lattice generated by the trailing submatrix $R(k:n, k:n)$ of R using Procedure 4. Moreover, since the LLL algorithm is incorporated into the algorithm, this procedure should also maintain the upper triangular structure of R and update the unimodular matrix Z. The following is an implementation of the procedure.

Procedure 5 (`Transform`$(k, \mathbf{z})$**[33]**) *Given $R \in \mathbb{R}^{n\times n}$ upper triangular, $Z \in \mathbb{Z}^{n\times n}$ unimodular, and an integer vector $\mathbf{z} \in \mathbb{Z}^{n-k+1}$, $1 \le k \le n-1$, such that* $\gcd(z_i) = \pm 1$,

```
1.  for $j = n - k + 1$ downto 2
2.      $M_j \leftarrow \texttt{Unim2}(z_{j-1}, z_j)$;
3.      $U_j \leftarrow \mathrm{diag}(I_{j-2}, M_j, I_{n-j-k+1})$;
4.      $\mathbf{z} \leftarrow U_j^{-1}\mathbf{z}$;
5.      $Z_j \leftarrow \mathrm{diag}(I_{k-1}, U_j)$;
6.      $R \leftarrow RZ_j$;
7.      $Z \leftarrow ZZ_j$;
8.      find a plane reflection $Q_j$ to restore the upper triangular structure of $R$;
9.      $R \leftarrow Q_jR$;
10. endfor
```

The major cost of the above procedure is in step 7, where the unimodular Z is updated. It requires $O(n)$ floating-point operations for each iteration. Thus the total cost is $O(n(n-k))$. In particular, when the size of $\mathbf{z}$ is n, the complexity is $O(n^2)$. Hence Procedure 5 is much more efficient than Algorithm 4.

6 New Algorithm for Computing Minkowski Reduced Bases: I

In 1985, Helfrich [17] and Afflerbach and Grothe [16] proposed algorithms for constructing the Minkowski reduced bases for lattices of arbitrary dimensions. The Helfrich's algorithm is based on the Kannan strategy [7]. Consequently, there are severe restrictions on applying the algorithm. Therefore this algorithm is of theoretical significance. On the other hand, the Afflerbach's algorithm is based on the Phost strategy [40]. It is practical for lattices of dimensions less than 8. For lattices of dimensions larger than 7, however, the algorithm may fail.

In this section, we present a new algorithm for constructing the Minkowski reduced bases for general lattices. The proposed new algorithm is based on Schnorr-Euchner strategy [26]. So it is much more efficient than the existing algorithms [17, 16] and it is practical for lattices of arbitrary dimensions. We first state the following result useful for our derivation.

Lemma 1 ([43]) *Let $B = [\mathbf{b}_1, \cdots, \mathbf{b}_n] \in \mathbb{R}^{m\times n}$ and L be the lattice generated by B. For a vector $\mathbf{v} = \sum_{i=1}^{n} v_i\mathbf{b}_i$ there exists a basis for L containing $\{\mathbf{b}_1, \cdots, \mathbf{b}_k, \mathbf{v}\}$ if and only if* $\gcd(v_{k+1}, \cdots, v_n) = 1$.

For clarity, as the HKZ algorithm in Section 5, we present a recursive version of our algorithm. Apparently, the first Minkowski reduced basis vector $\mathbf{m}_1 = B\mathbf{z}$ is a shortest nonzero lattice vector in L, which can be obtained by applying the sphere decoding algorithm [1]. We can extend $\mathbf{m}_1$ to a basis for L by calling Procedure $\texttt{Transform}(1, \mathbf{z})$. Now, suppose that a basis $\{\mathbf{m}_1, \cdots, \mathbf{m}_{p-1}, \mathbf{b}'_p, \cdots, \mathbf{b}'_n\}$, $1 < p \le n$, has been obtained, to extend $\{\mathbf{m}_1, \cdots, \mathbf{m}_{p-1}\}$ to a Minkowski reduce basis for L, we must solve the following two problems:

- Constructing the pth Minkowski reduced basis vector $\mathbf{m}_p$.
- Extending $\{\mathbf{m}_1, \cdots, \mathbf{m}_p\}$ to a basis for L.

To solve the first problem, denoting $B_p = [\mathbf{m}_1 \ \cdots \ \mathbf{m}_{p-1} \ \mathbf{b}'_p \ \cdots \ \mathbf{b}'_n]$, from Lemma 1, we have

$$\|\mathbf{m}_p\|_2 = \min\{\|B_p\mathbf{z}\|_2 : \ \mathbf{z} = [z_1, \cdots, z_n]^{\mathrm{T}} \in \mathbb{Z}^n, \ \gcd(z_p, \cdots, z_n) = 1\}.$$

The above minimization problem can be viewed as the shortest nonzero lattice vector problem with the constraint $\gcd(z_p, \cdots, z_n) = 1$. Thus $\mathbf{m}_p$ can be obtained by modifying the sphere decoding algorithm [1]. Instead of using the length of the first column of the lattice generating matrix as the initial size of search region, we use the length of the pth column, since $\mathbf{z} = \mathbf{e}_p$, the pth unit vector, satisfies the constraint $\gcd(z_p, \cdots, z_n) = 1$. The LLL algorithm can be incorporated into the following algorithm as a preprocessor to accelerate the search process.

Algorithm 6 (`M-Decode-1`(R, p)) *Given an upper triangular lattice generating matrix $R \in \mathbb{R}^{n\times n}$ and an integer $1 \le p < n$, this algorithm finds a solution vector $\mathbf{z} = [z_1, \cdots, z_n]^{\mathrm{T}} \in \mathbb{Z}^n$ for the following constrained optimization problem:*

$$\min \|Rx\|_2 \quad \textit{such that} \quad x \in \mathbb{Z}^n \ \textit{and} \ \gcd\{x_p, \cdots, x_n\} = 1. \tag{13}$$

```
1.  if $n = 1$, return $\mathbf{z} = 1$; endif
2.  Set the initial size $\rho \leftarrow \|R(:,p)\|_2$ of the search region;
3.  Call M-Search-1($R$, $\mathbf{0}$, [], 0, $\rho$, $p$);
```

As shown above, this is a wrapper function. It calls `M-Search-1`, which finds a solution for a more general problem, the closest point search problem with the constraint $\gcd(z_p, ..., z_n) = 1$. We present a recursive version of the procedure.

Procedure 6 (`M-Search-1`(R, $\mathbf{y}$, $\mathbf{z}_{\mathrm{in}}$, l, ρ, p)) *Let $R \in \mathbb{R}^{n\times n}$ be an upper triangular lattice generating matrix, the center $\mathbf{y} \in \mathbb{R}^n$ and the size ρ of the search region. This procedure finds an integer vector $\mathbf{z}$ minimizing $\|R\mathbf{z} - \mathbf{y}\|_2$, subject to the constraint $\gcd(z_p, ..., z_n) = 1$. The parameter $\mathbf{z}_{\mathrm{in}}$ is a partial solution and l is the distance between the current lattice vector and the corresponding sublattice.*

```
1.  Initialize $\mathbf{z}$ to an empty vector;
2.  for each index $u_n$ in (10)
3.      Update the distance $l_n = \sqrt{l^2 + (y_n - u_n R(n,n))^2}$;
4.      if $l_n < \rho$
5.          Build solution bottom up: $\hat{\mathbf{z}} = [u_n \ \mathbf{z}_{\mathrm{in}}]^{\mathrm{T}}$;
6.          if recursion level 1
7.              if (ggcd($\hat{\mathbf{z}}_{p:n}$) = 1)
8.                  Reset $\rho \leftarrow l_n$;
9.                  Save $\mathbf{z} = \hat{\mathbf{z}}$;
10.             endif
11.         else
12.             Update the center $\hat{\mathbf{y}} = \mathbf{y}_{1:n-1} - u_n R(1:n-1, n)$;
13.             Call M-Search-1($R(1:n-1, 1:n-1)$, $\hat{\mathbf{y}}$, $\hat{\mathbf{z}}$, $l_n$, $\rho$, $p$);
14.             if $\tilde{\mathbf{z}}$ returned above is non-empty, set $\mathbf{z} = \tilde{\mathbf{z}}$; endif
15.         endif
16.     endif
17. endfor
18. Return $\mathbf{z}$.
```

Note that in lines 7 and 8, whenever a shorter lattice vector satisfying the constraint is found, the size ρ is reduced. When the LLL algorithm is incorporated at the beginning, the unimodular matrix obtained from the LLL algorithm should be applied to $\hat{\mathbf{z}}$ when the greatest common divisor is calculated in line 6. The function `ggcd` in line 6 is a generalization of the greatest common divisor (gcd) of two integers.

Procedure 7 (`ggcd(a)`) *This function returns the greatest common divisor of the entries of an integer n-array $\mathbf{a}$, $n \geq 1$.*

```
1. $d = 0$;
2. for $i = 1$ to $n$
3.     $d = \gcd(d, a_i)$;
4.     if $d = 1$ break;
5. endfor
6. return $d$.
```

Once the pth Minkowski reduced basis vector $\mathbf{m}_p = B_p\mathbf{z}$ is found by Algorithm 6, the second problem is to extend $(\mathbf{m}_1, \cdots, \mathbf{m}_p)$ to a basis for L. In terms of matrices, it is to find a unimodular matrix Z such that

$$B_{p+1} = B_p Z, \tag{14}$$

which implies that the first $p-1$ columns of Z are the first $p-1$ unit vectors $\mathbf{e}_i$, $i = 1, ..., p-1$ and the pth column of Z is $\mathbf{z} = [z_1 \ ... \ z_n]^{\mathrm{T}}$ found by Algorithm 6, so that the first $p-1$ columns of B_{p+1} equal the first $p-1$ columns $\mathbf{m}_1, ..., \mathbf{m}_{p-1}$ of B_p and the pth column of B_{p+1} is $\mathbf{m}_p = B_p\mathbf{z}$ as desired. Since $\gcd(z_p, ..., z_n) = 1$, from Section 5, we can construct a unimodular matrix M_p whose first column is $[z_p \ ... \ z_n]^{\mathrm{T}}$. Now consider the two unimodular matrices

$$Z_1 = \begin{bmatrix} I_{p-1} & 0 \\ 0 & M_p \end{bmatrix} \quad \text{and} \quad Z_2 = \begin{bmatrix} 1 & & & z_1 & & \\ & \ddots & & \vdots & & \mathbf{0} \\ & & 1 & z_{p-1} & & \\ & & & 1 & & \\ & \mathbf{0} & & & \ddots & \\ & & & & & 1 \end{bmatrix}. \tag{15}$$

We claim that the product Z_1Z_2 is a unimodular matrix satisfying (14). Indeed, Z_1Z_2 is unimodular since both Z_1 and Z_2 are unimodular, from (15), the first $p-1$ columns of Z_1Z_2 are the first $p-1$ unit vectors and the pth column of Z_1Z_2 is $\mathbf{z} = [z_1 \ ... \ z_n]^{\mathrm{T}}$.

The application of Z_1 can be performed by Procedure 5 and the application of Z_2 is the calculation of a linear combination of the first p columns.

The new algorithm for constructing a Minkowski reduced basis for a general lattice is summarized as follows. Again, the LLL algorithm can be incorporated into `M-Decode-1` as a preprocessor to improve the performance.

Algorithm 7 (`M-Red-1`(B)) *Given a lattice generating matrix $B \in \mathbb{R}^{m \times n}$, $m \geq n$, this algorithm computes a unimodular matrix Z such that the columns of BZ form a Minkowski reduced basis.*

```
1.  QR decomposition: $B = QR$;
2.  $Z \leftarrow I_n$;
3.  for $k = 1$ to $n$
4.      Call M-Decode-1($R$, $k$) to find the solution $\mathbf{z}$ for (13);
5.      Call Transform($k, [z_k, \cdots, z_n]^{\mathrm{T}}$) to apply $Z_1$ in (15);
6.      Apply $Z_2$ in (15) to $R$:
        $R(1:k-1,k) \leftarrow R(1:k-1,k) + R(1:k-1,1:k-1)[z_1,\cdots,z_{k-1}]^{\mathrm{T}}$.
7.      Apply $Z_2$ to $Z$:
        $Z(:,k) \leftarrow Z(:,k) + Z(:,1:k-1)[z_1,\cdots,z_{k-1}]^{\mathrm{T}}$.
8.  endfor
```

7 New Algorithm for Computing Minkowski Reduced Bases: II

From the brief analysis in Section 6, we can see that Algorithm 6 usually costs more than the sphere decoding algorithm [1], because the search size ρ is usually less often reduced than the sphere decoding algorithm due to the constraint in (13), or the condition $\gcd(\hat{\mathbf{z}}_{p:n}) = 1$ in line 6 of Procedure 6. This motivates us to improve Algorithm 6. Our idea is to impose the constraint as early as possible to reduce the number of points to be searched. Clearly, $\gcd(\mathbf{z}_{p:n})$ can be calculated as soon as $z_p, ..., z_n$ are available. In Procedure 6, a solution is built bottom-up, from z_n to z_1, thus the condition $\gcd(\mathbf{z}_{p:n}) = 1$ can be checked at level p, instead of level 1 as in Procedure 6. Here is the modified Procedure 6 based on the above idea.

Procedure 8 (M-Search-2(R, $\mathbf{y}$, $\mathbf{z}_{\text{in}}$, l, ρ, p)) *Let $R \in \mathbb{R}^{n\times n}$ be an upper triangular lattice generating matrix, the center $\mathbf{y} \in \mathbb{R}^n$ and the size ρ of the search region. This procedure finds an integer vector $\mathbf{z}$ minimizing $\|R\mathbf{z} - \mathbf{y}\|_2$, subject to the constraint $\gcd(z_p, ..., z_n) = 1$. The parameter $\mathbf{z}_{\text{in}}$ is a partial solution and l is the distance between the current lattice vector and the corresponding sublattice.*

```
1.   Initialize $\mathbf{z}$ to an empty vector;
2.   for each index $u_n$ in (10)
3.       Update the distance $l_n = \sqrt{l^2 + (y_n - u_n R(n,n))^2}$;
4.       if $l_n < \rho$
5.           Build solution bottom up:    $\hat{\mathbf{z}} = [u_n\ \mathbf{z}_{\text{in}}]^{\mathrm{T}}$;
6.           if recursion level 1
7.               if $(l_n > 0)$
8.                   Reset $\rho \leftarrow l_n$;
9.                   Save $\mathbf{z} = \hat{\mathbf{z}}$;
10.              endif
11           else
12.              if $(n \neq p)$ or ($n = p$ andggcd($\hat{\mathbf{z}}_{p:n}$) $= 1$)
13.                  Update the center $\hat{\mathbf{y}} = \mathbf{y}_{1:n-1} - u_n R(1:n-1,n)$;
14.                  Call M-Search-2 ($R(1:n-1,1:n-1)$, $\hat{\mathbf{y}}$, $\hat{\mathbf{z}}$, $l_n$, $\rho$, $p$);
15.                  if $\tilde{\mathbf{z}}$ returned above is non-empty, set $\mathbf{z} = \tilde{\mathbf{z}}$; endif
16.              endif
17.          endif
```

```
18.       endif
19. endfor
20. Return $\mathbf{z}$.
```

As shown above, the $(p-1)$-dimensional sublattices indexed by those $\hat{z}_p$ not satisfying $\gcd(\hat{\mathbf{z}}_{p:n}) = 1$ are excluded from the search. Consequently, the number of lattice points to be searched is significantly reduced.

Now we have the second algorithm for constructing the Minkowski reduced basis by simply replacing `M-Search-1` in Algorithm 6 with the above `M-Search-2`.

Algorithm 8 (`M-Decode-2`(R, p)) *Given an upper triangular lattice generating matrix $R \in \mathbb{R}^{n\times n}$ and an integer $1 \le p < n$, this algorithm finds a solution vector $\mathbf{z} = [z_1, \cdots, z_n]^{\mathrm{T}} \in \mathbb{Z}^n$ for the constrained optimization problem 13.*

```
1.  if $n = 1$, return $\mathbf{z} = 1$; endif
2.  Set the initial size $\rho \leftarrow \|R(:,p)\|_2$ of the search region;
3.  Call M-Search-2($R$, $\mathbf{0}$, [], 0, $\rho$, $p$);
```

Procedure 8 improves Procedure 6, with the caveat that the LLL algorithm cannot be used as its preprocessor to accelerate the search process. As pointed out early, when the LLL algorithm is used as a preprocessor, the unimodular matrix obtained from the LLL algorithm must be applied to solution vector $\mathbf{z}$ when $\gcd(\mathbf{z}_{p:n})$ is calculated. However, the application of the unimodular matrix obtained from the LLL algorithm requires a complete n-vector, whereas at level p, where $\gcd(\mathbf{z}_{p:n})$ is calculated in Procedure 8, only a subvector $\mathbf{z}_{p:n}$ is available. To alleviate the problem, we propose a new technique for accelerating Procedure 8 by introducing the definition of the quasi-LLL reduced basis.

Consider an $n \times n$ unimodular matrix Z with the following structure:

$$Z = \begin{pmatrix} D & E \\ 0^{(n-p+1)\times(p-1)} & F \end{pmatrix}, \tag{16}$$

where D, E, and F have proper dimensions. Then both D and F are unimodular. If an integer vector $\hat{\mathbf{z}}$ satisfies $\gcd(\hat{\mathbf{z}}_{p:n}) = 1$ then the integer vector $\mathbf{z} = Z\hat{\mathbf{z}}$ also satisfies the condition $\gcd(\mathbf{z}_{p:n}) = 1$, since $\mathbf{z}_{p:n}^{\mathrm{T}} = F\hat{\mathbf{z}}_{p:n}^{\mathrm{T}}$ and F is unimodular. Thus we propose a notion of reduced basis, called quasi-LLL reduced basis by imposing the structure (16).

Definition 5 (quasi-LLL(p) reduced) *Given an $\omega \in (0.25, 1.0)$ and an integer p: $1 \le p \le n$, a lattice basis $\{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_n\}$ is called quasi-LLL(p) reduced if the upper triangular factor R in the decomposition (1) of $B = [\mathbf{b}_1\ \mathbf{b}_2\ ...\ \mathbf{b}_n]$ satisfies*

$$|r_{i,j}| \le |r_{i,i}|/2, \quad 1 \le i < j \le n \quad \textit{(size-reduced)}, \tag{17}$$

and

$$r_{i,i}^2 + r_{i-1,i}^2 \ge \omega\, r_{i-1,i-1}^2, \quad i = 2, 3, \cdots, p-1, p+1, \cdots, n. \tag{18}$$

The only difference between Definition 4 and Definition 5 is that a quasi-LLL reduced basis may not satisfy the condition

$$r_{p-1,p}^2 + r_{p,p}^2 \ge r_{p-1,p-1}^2. \tag{19}$$

In particular, when $p = 1$, a quasi-LLL reduced basis is LLL reduced.

Now, we present an algorithm for computing a quasi-LLL reduced basis.

Algorithm 9 (`quasi-LLL`(B, p, ω)) *Given a lattice generating matrix $B = [\mathbf{b}_1, \cdots, \mathbf{b}_n] \in \mathbb{R}^{m\times n}$ and parameters $\omega \in \mathbb{R}$ and $p \in \mathbb{Z}$, where $0.25 < \omega < 1$ and $1 \le p \le n$. This algorithm computes an upper triangular matrix $R \in \mathbb{R}^{n\times n}$ and a unimodular matrix $Z \in \mathbb{Z}^{n\times n}$ of the structure (16), such that the columns of BZ form a quasi-LLL reduced basis.*

```
1.  Compute the QR decomposition: $B = QR$;
2.  Set $Z \leftarrow I$;
3.  Set $k \leftarrow 2$;
4.  while $k < p$
5.      Reduce($k-1, k$);
6.      if $r_{k,k}^2 + r_{k-1,k}^2 < \omega\, r_{k-1,k-1}^2$;
7.          SwapRestore($k$);
8.          $k \leftarrow \max(k-1, 2)$;
9.      else
10.         for $i = k-2$ downto 1
11.             Reduce($i, k$);
12.         endfor
13.         $k \leftarrow k+1$;
14.     endif
15. endwhile
16. for $i = p-1$ downto 1
17.     Reduce($i, p$);
18. endfor
19. $k \leftarrow p+1$;
20. while $k \le n$
21.     Reduce($k-1, k$);
22.     if $r_{k,k}^2 + r_{k-1,k}^2 < \omega\, r_{k-1,k-1}^2$;
23.         SwapRestore($k$);
24.         $k \leftarrow \max(p+1, k-1)$;
25.     else
26.         for $i = k-2$ downto 1
27.             Reduce($i, k$);
28.         endfor
29.         $k \leftarrow k+1$;
30.     endif
31. endwhile
```

As shown above, two parts, lines 3–15 and lines 19–31, of the algorithm respectively apply the LLL algorithm to two separate blocks of R. When $p = 1$, this algorithm is the same as the LLL algorithm. Now we can incorporate Algorithm 9 into Algorithm 8 to improve its performance:

Algorithm 10 (`LLL-aid-M-Decode-2`(R, p, ω)) *Given an upper triangular matrix $R \in \mathbb{R}^{n\times n}$ and an integer $1 \le p \le n$, this algorithm computes a vector $\mathbf{z} = [z_1, \cdots, z_n]^{\mathrm{T}} \in \mathbb{Z}^n$ so that $R\mathbf{z}$ is a shortest lattice point with* $\gcd(z_p, \cdots, z_n) = 1$.

```
1.  $[R,\ Z] \leftarrow$ quasi-LLL($R,\ p,\ \omega$);
```

```
2.  $\hat{\mathbf{z}} \leftarrow$ M-Decode-2($R$, $p$);
3.  Return $\mathbf{z} \leftarrow Z\,\hat{\mathbf{z}}$.
```

Finally, simply replacing `M-Decode-1` in line 4 of Algorithm 7 with the above algorithm, we have the second algorithm for constructing a Minkowski reduced basis for a given lattice.

Algorithm 11 (`LLL-aid-M-Red-2`(B, ω)) *Given a lattice generating matrix $B \in \mathbb{R}^{m\times n}$, $m \geq n$, this algorithm computes a unimodular matrix Z such that the columns of BZ form a Minkowski reduced basis.*

```
1.  QR decomposition: $B = QR$;
2.  $Z \leftarrow I_n$;
3.  for $k = 1$ to $n$
4.      Call LLL-aid-M-Decode-2($R$, $k$, $\omega$) to find the solution $\mathbf{z}$ for (13);
5.      Call Transform($k, [z_k, \cdots, z_n]^{\mathrm{T}}$) to apply $Z_1$ in (15);
6.      Apply $Z_2$ in (15) to $R$:
        $R(1:k-1,k) \leftarrow R(1:k-1,k) + R(1:k-1,1:k-1)[z_1,\cdots,z_{k-1}]^{\mathrm{T}}$.
7.      Apply $Z_2$ to $Z$:
        $Z(:,k) \leftarrow Z(:,k) + Z(:,1:k-1)[z_1,\cdots,z_{k-1}]^{\mathrm{T}}$.
8.  endfor
```

8 Concluding Remarks

This paper presents three practical lattice basis reduction algorithms. In Section 5, a new HKZ lattice reduction algorithm is presented. A unimodular transformation is used in our new algorithm. Our analysis shows that our algorithm is significantly more efficient than the Kannan's strategy. Also, two algorithms for computing the Minkowski reduced basis are proposed. In Section 6, we illustrate the ideas of our algorithms and give the first practical algorithm for computing the Minkowski reduced basis for a lattice. The LLL algorithm can be incorporated into both our HKZ lattice basis reduction algorithm and the first Minkowski lattice basis reduction algorithm to improve their performance. In Section 7, we present the second Minkowski lattice basis reduction algorithm, which significantly improves our first Minkowski reduction algorithm by drastically reducing the number of lattice points being searched. Furthermore, to improve its performance, we propose a quasi-LLL reduction technique.

References

[1] Erik Agrell, Thomas Eriksson, Alexander Vardy, and Kenneth Zeger. Closest point search in lattices. *IEEE Transactions on Information Theory*, vol. 48, no. 8, 2002, 2201–2214.

[2] A. Barvinok. *A Course in Convexity.* Graduate Studies in Mathematics 54. American Mathematical Society, Providence, RI, 2002.

[3] J.W.S. Cassels. *An Introduction to the Geometry of Numbers, Second Printing.* Springer-Verlag, Berlin, Heidelberg, 1997.

[4] H. Cohen. *A Course in Computational Algebraic Number Theory.* Graduate Texts in Mathematics 138, Second corrected printing. Springer, Berlin, 1995.

[5] G.H. Golub and C.F. Van Loan. *Matrix Computations, Third Edition.* The Johns Hopkins University Press, Baltimore, MD, 1996.

[6] A. Joux and J. Stern. Lattice reduction: A toolbox for the cryptanalyst. *Journal of Cryptology*, 11(**3**), 1998, 161–185.

[7] Ravi Kannan. Improved algorithms for integer programming and related lattice problems. *Proc. ACM Symp. Theory of Computing*, Boston, MA, Apr, 1983, 193–206.

[8] Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 3(**12**), 1987, 415–440.

[9] A. Korkine and G. Zolotareff. Sur les formes quadratiques. *Maht. Ann.*, **6**, 1873, 366–389.

[10] A.K. Lenstra, H.W. Lenstra, Jr. and L. Lovász. Factorizing polynomials with rational coefficients. *Mathematicsche Annalen*, **261**, 1982, 515–534.

[11] F.T. Luk and D.M. Tracy. An improved LLL algorithm. *Linear Algebra and its Applications*, **428**(2–3), 2008, 441–452.

[12] H. Minkowski. *Geometrie der Zahlen.* Teubner, Leipzig, 1896.

[13] The LLL Algorithm: Survey and Applications. *Information Security and Cryptography, Texts and Monographs.* Editors Phong Q. Nguyen and Brigitte Vallée. Springer Heidelberg Dordrecht London New York, 2010.

[14] H. Minkowski. Über die positiven quadratischen Formen und über kettenbruchähnliche Algorithmen. *Journal fur die Reine und Angewandte Mathematik*, vol. 107, 1891, 278–297.

[15] J. C. Lagarias. Worst-case complexity bounds for algorithms in the theory of integral quadratic forms. *J. Algorithms*, **1**, 1980, 142–186.

[16] L. Afflerbach and H. Grothe. Calculation of Minkowski-reduced lattice bases. *Computing*, vol. 35, no. 3-4, 1985, 269–276.

[17] Bettina Helfrich. Algorithms to construct Minkowski reduced and Hermite reduced lattice bases. *Theoretical Computer Science*, vol. 41, no. 2-3, 1985, 125–139.

[18] D. E. Knuth. *The Art of Computer Programming*, 2nd ed. Reading, MA: Addison-Wesley, 1981, vol. 2.

[19] G. Marsaglia. The structure of linear congruential sequences. *Applications of Number Theory to Numerical Analysis (S. K. Zaremba, ed)*, 1972, 249-285.

[20] W. A. Beyer, R. B. Roof, and D. Williamson. The lattice structure of multiplicative congruential pseudo-random vectors. *Math. Comput*, **25**, 1971, 345-360.

[21] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, Spriger, Berlin, 1993.

[22] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: A Cryptographic Perspective*, Kluwer Internat. Ser. Engrg. Comput. Sci. 671, Kluwer Academic Publishers, Boston, MA, 2002.

[23] Y. H. Gan, C. Ling, and H. M. Mow. Complex lattice reduction algorithm for low-complexity full-diversity MIMO detection. *IEEE Transactions on Signal Processing*, vol. **57**, no. 7, 2009, 2701-2710.

[24] D. Boneh. Twenty years of attacks on the RSA cryptosystem. *Notices Amer. Math. Soc*, **46**, 1999, 203–213.

[25] C. Hermite. Extraits de lettres de M. Hermite *à* M. Jacobi sur diff*é*rents objets de la th*é*orie des nombres. *J. Reine Angew. Math*, **40**, 1850, 279–290.

[26] C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Programming*, **66**, 1994, 181–199.

[27] A. H. Banihashemi and A. K. Khandani. On the complexity of decoding lattices using the Korking-Zolotarev reduced basis. *IEEE Trans. Inform. Theory*, **44**, 1998, 162–171.

[28] D. Micciancio. The hardness of the closest vector problem with preprocessing. *IEEE Trans. Inform. Theory*, **47**, 2001, 1212–1215.

[29] S. Arora, L. Babai, J. Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. Syst. Sci*, **54**, 1997, 317–331.

[30] X. W. Chang and G. H. Golub. Solving ellipsoid-constrained integer least squares problems. *SIAM J. Matrix Anal. Appl*, **31**, 3, 2009, 1071–1089.

[31] C. P. Schnorr. A hierarchy of polynomial lattice basis reduction algorithms. *Theo. Comput. Sci*, **53**, 1987, 201–224.

[32] P. Q. Nguyen and D. Stehl*é*. An LLL algorithm with quadratic complexity. *SIAM J. Comput*, **39**, 3, 2009, 874–903.

[33] F.T. Luk, S. Qiao, and W. Zhang. A Lattice Basis Reduction Algorithm. *Institute for Computational Mathematics Technical Report 10-04. Hong Kong Baptist University.*

[34] F.T. Luk and S. Qiao. A Pivoted LLL Algorithm. *Linear Algebra Appl*, 2010.

[35] M. O. Damen, H. E. Gamal, and G. Caire. On maximum-likelihood detection and the search for the closest lattice point. *IEEE Trans. Inf. Theory*, **49**, 2003, 2389–2402.

[36] M. Taherzadeh, A. Mobasher, and A. K. Khandani. LLL reduction achieves the receive diversity in MIMO decoding. *IEEE Trans. Inf. Theory*, **53**, 2007, 4801–4805.

[37] A. Vardy and Y. Be'ery. Maximum-likelihood decoding of the Leech lattice. *IEEE Trans. Inform. Theory*, **39**, 1993, 1435–1444.

[38] A. H. Banihashemi and I. F. Blake. Trellis complexity and minimal trellis diagrams of lattices. *IEEE Trans. Inform. Theory*, **44**, 1998, 1829–1847.

[39] U. Fincke and M. Pohst. Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. *Math. of Comput*, **44**, 1985, 463–471.

[40] M. Phost. On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications. *ACM SIGSAM Bull*, **15**, 1981, 37–44.

[41] E. Viterbo and J. Boutros. A universal lattice code decoder for fading channels. *IEEE Trans. Inform. Theory*, **45**, 1999, 1639–1642.

[42] L. Babai. On Lovász's lattice reduction and the nearest lattice point problem. *Combinatorica*, **6**, 1986, 1–13.

[43] B.L. van der Waerden and H. Gross. *Studien zur Theorie der Quadratischen Formen.* Birkhäuser, Basel, 1968.